UNIVERSIDAD CARLOS III DE MADRID
ESCUELA POLITECNICA SUPERIOR

Departamento de Tecnología Electrónica
INGENIERÍA TÉCNICA INDUSTRIAL
ESPECIALIDAD ELECTRÓNICA INDUSTRIAL

PROYECTO FIN DE CARRERA

DESARROLLO DE UNA INTERFAZ DE TECLADO PS/2 ESTANDAR CON PANTALLA VGA.

Autor: Dairon Lamas Duarte
Tutor: Celia López Ongil
Curso: 2012-2013

1. Agradecimientos.

Primordialmente quiero agradecer a la universidad Carlos III de Madrid, puesto que me brindó conocimientos con los cuales pude desarrollar todo el proyecto y saber afrontar todos los problemas y dificultades durante la elaboración final de este.

A los profesores que me brindaron su sabiduría y conocimiento constantemente ante cualquier aspecto o pregunta, ayudándome así a subsanar las dificultades que se fueron presentando durante el desarrollo del proyecto.

No puedo dejar de dar las gracias a todos y cada uno de los compañeros de clase que de varias maneras siempre estuvieron acompañándome, apoyándome, guiándome y ayudándome en los momentos que requería ayuda. Agradecer por compartir conocimientos conmigo y por dejaran muchas enseñanzas y experiencias.

2. Índice de Contenido.

3. Índice de Figuras.

4. Índice de Tablas.

5. Dedicatoria.

Principalmente quiero dedicar este trabajo a mis padres, ya que me brindaron todo el apoyo y la fortaleza necesaria en el desarrollo y transcurso de éste; ayudándome a concluirlo satisfactoriamente y en muchas ocasiones a guiarme y animarme para seguir adelante.

Dedico también el proyecto a mi tutora, la profesora Celia López Ongil; por haber depositado toda su confianza en mí y por haber tenido tanta paciencia y compresión con mis dudas y problemas. En todo momento me brindó su sabiduría, amor y tiempo; lo que se convirtió en un factor positivo a mi favor en todo momento.

Me gustaría también dedicar este proyecto a todas aquellas personas que de una forma u otra han tenido que ver con la realización del mismo. A todas esas personas que directa o indirectamente colaboraron conmigo y con mi trabajo día a día. Tanto amigos, conocidos, profesores, compañeros de trabajo y personas que me han facilitado alguna cosa, implicándose todos con el desarrollo de este mi proyecto final de carrera.

"No tenemos la oportunidad de hacer muchas cosas, por lo que cada cosa que hagamos debe ser excelente. Porque esta es nuestra vida".

Steve Jobs

6. Introducción y Motivación.

El diseño y prototipado de circuitos digitales de complejidad media por parte de los estudiantes de grado y posgrado supone un esfuerzo notable en términos de aprendizaje no sólo de metodologías de diseño sino de herramientas CAD comerciales. Si bien la depuración de los circuitos debe realizarse siempre, en las primeras etapas, mediante simulación de las descripciones de los circuitos, en las últimas etapas del ciclo de diseño, el prototipado aportará la prueba definitiva de funcionamiento correcto del circuito especificado. Este prototipado y su validación funcional suponen una dificultad añadida cuando las entradas y salidas del bloque diseñado no son fáciles de generar o de observar. La creación y mantenimiento de una biblioteca de componentes de entrada/salida estándar (teclados, pantallas, ratones, etc.) supondrá una ayuda notable para los trabajos de grado y posgrado en el campo de la electrónica digital.

Este proyecto aborda el estudio, diseño e implementación de un conjunto de bloques digitales, que permiten la interfaz con elementos de Entrada/Salida de datos para sistemas digitales. Así, se generan dos bloques de interfaz, con un teclado *PS/2* y con una pantalla *VGA*, y se conectan entre sí mediante otro bloque controlador para permitir la visualización gráfica en una pantalla *VGA* de las teclas pulsadas en un teclado *PS/2*. Este circuito se ha prototipado en un dispositivo programable *FPGA* (*Field Programmable Gate Array* o Campo de Matriz de Puertas Programables) que está contenido en una placa de desarrollo de *Xilinx* (*Spartan 3E Starter Board*).

El diseño y desarrollo de unos bloques que permitan realizar de forma rápida y eficiente la interfaz con dispositivos de Entrada/Salida (teclado *PS/2* y pantalla *VGA*), se plantea en el departamento de Tecnología Electrónica para dar un conjunto de herramientas básicas, en los trabajos y prácticas desarrollados en las distintas asignaturas impartidas, para realizar circuitos de complejidad media-alta. Además, se pretende facilitar el prototipado rápido de los futuros diseños y, por ello, se impone la necesidad de desarrollar estos bloques en un formato portable, reutilizable y sintetizable en distintas tecnologías; esto es un lenguaje de descripción de hardware (*VHDL*) y con descripciones sintetizables.

El conjunto de bloques desarrollados serán una biblioteca de componentes de utilidad fundamentalmente educativa. La principal aplicación de los mismos es probar, analizar y depurar futuros circuitos o sistemas digitales. Además, permitirá a los futuros estudiantes de grado y posgrado realizar circuitos de mayor complejidad y depurar su funcionamiento de forma más rápida y eficiente; ya sea para trabajos o prácticas de laboratorio de las asignaturas impartidas por el departamento de Tecnología Electrónica.

7. Objetivos.

En este proyecto Fin de Carrera se plantea desarrollar un conjunto de bloques digitales que permitan la comunicación con dispositivos de Entrada/Salida de datos en sistemas digitales. Los bloques a diseñar son la interfaz con un teclado tipo *PS/2* estándar, la interfaz con una pantalla *VGA* estándar y un bloque que comunique y gestione ambas interfaces. Para probar estos bloques se incluirán en un circuito de mayor jerarquía que permite dibujar en una pantalla *VGA* las teclas pulsadas en el teclado, reconociendo y almacenando dichas teclas. De esta forma el sistema general quedaría compuesto por cuatro bloques de funcionamiento individual:

- Bloque 1: Interfaz y protocolo de comunicación del teclado *PS/2*.
- Bloque 2: Interfaz y protocolo de comunicación de la pantalla *VGA*.
- Bloque 3: Circuito de control y trazado gráfico de la letra pulsada [DRIVER].
- Bloque 4: Conexión de los bloques 1, 2 y 3.

Un teclado *PS/2* convencional normalmente cuenta con 102 teclas, sin contar las teclas multimedia en los teclados extendidos modernos; cuenta con un conector *PS/2* para comunicarse con el sistema digital que procese las teclas pulsadas.

Una pantalla o monitor *VGA* convencional, es un dispositivo gráfico que representa la información procedente del sistema digital que la genera. Dicha pantalla tiene como entradas la alimentación y un conector con 5 pines (Colores *R-G-B*, así como el blanqueo de las filas y de las columnas).

El funcionamiento global del circuito compuesto por los 4 bloques individuales anteriormente especificados, consiste en pulsar cualquier tecla en un teclado convencional tipo *PS/2*; ésta deberá ser leída, interpretada y almacenada en la *FPGA.* Luego de ser procesada se envié a la pantalla *VGA* y se dibuje en pantalla dicha letra o carácter pulsado. Partiendo de la funcionalidad general se definen unos objetivos específicos para el desarrollo del proyecto, los cuales son:

- Diseñar e implementar la interfaz y el protocolo de comunicación del teclado *PS/2* estándar, para que se reciba la tecla presionada y se almacene para utilizarla en posteriores procesos.
- Diseñar e implementar la interfaz y el protocolo de comunicación de la pantalla *VGA*, para poder dibujar en la pantalla y utilizar colores.
- Diseñar e implementar un circuito de control y de trazado gráfico de la letra pulsada [DRIVER], que decodifique la tecla recibida del teclado *PS/2* y la dibuje en la pantalla *VGA* hasta que se presione otra tecla o se realice una inicialización asíncrona (*Reset*).
- Lograr la comunicación entre todos los bloques, para cumplir con la funcionalidad global.

8. Estado de la técnica.

8.1. Acerca de la FPGA.

Una *FPGA* (*Field Programmable Gate Array* o Matriz de Puertas Programable por Campo) es un dispositivo semiconductor que contiene bloques de lógica que se pueden conectar, combinar y ser configurados con una funcionalidad especificada previamente. Estos bloques pueden realizar funciones combinacionales o secuenciales. Además, las *FPGAs* pueden contener bloques embebidos de muy alta complejidad, como *DSPs*, Microprocesadores, Memorias *SRAM*, *DRAM*, etc. Las *FPGAs* pueden contener en torno a cientos de miles de puertas lógicas equivalentes (*NAND*).

Las *FPGAs* tienen grandes ventajas en el campo industrial, lo que despierta mucho interés como herramienta y hace que se convierta en una de las opciones preferidas y utilizadas en el desarrollo de circuitos o dispositivos electrónicos digitales. Entre las ventajas que tienen se puede mencionar:

- Son dispositivos reprogramables, esto añade una enorme flexibilidad en el diseño de cualquier aplicación o circuito electrónico.
- Sus costes de desarrollo y adquisición son muy reducidos cuando se usan para pequeñas tiradas y el tiempo de desarrollo es también menor que para circuitos de aplicación específica (*ASIC*).
- Son muy versátiles, ya que hay modelos específicos para determinados usos.
- Son chips que presentan una robustez muy sólida y mucha fiabilidad en el diseño, hay que tener en cuenta que existen muchas herramientas a nivel de software para simular, hacer síntesis, prototipar e implementar cualquier diseño.

Según datos históricos las *FPGAs* surgen como una evolución de los conceptos desarrollados en las *PAL* (*Programmable Array Logic* o Lógica de Matriz Programable) y los *CPLD* (*Complex Programmable Logic Device* o Dispositivo Lógico Programable Complejo). Las *FPGAs* nacen en el año 1985 gracias a la empresa *Xilinx* cuyos fundadores fueron Ross Freeman y Bernard Vonderschmitt. (1) (2) (3)

Estos dispositivos representan hoy en día una alternativa interesante a la hora de realizar un diseño, debido a que permiten realizar un circuito a medida y sin elevados costes en comparación a otras tecnologías menos flexibles. En la actualidad podemos encontrar la presencia de dispositivos *FPGAs* en muchos sectores de la industria, desde sistemas de automoción, electrodomésticos, plantas automatizadas, etc.

Cualquier diseño digital puede ser implementado en un *FPGA*, siempre y cuando, ésta disponga de los recursos necesarios. Las aplicaciones donde el *DSP* (*Digital Signal Processing* o Procesamiento Digital de Señales) es la base principal, es donde más comúnmente se utilizan los *FPGAs*. Ejemplos de este tipo de aplicaciones son: seguridad electrónica (arcos de seguridad de bancos, alarmas), control industrial (control de instalaciones eléctricas, seguimiento de procesos automatizados) y equipos de medida (audio, potencia, transmisión de televisión), prototipos de *ASIC* (*Application Specific Integrated Circuit* o Circuito Integrado de Aplicación Específica), sistemas de imágenes para medicina, sistemas de visión para computadoras, reconocimiento de voz, emulación de hardware de computadora, etc.

El uso en otras áreas se está incrementando cada día más; en las aplicaciones que requieren un alto grado de paralelismo, donde se requieren múltiples interfaces digitales o muchas entradas y salidas, como las comunicaciones de alta velocidad de transmisión de datos; es más notable aún. El primer y mayor fabricante de estos dispositivos, es la compañía *Xilinx*. Otros fabricantes de estos dispositivos son *Altera*, *Microsemi* y *Atmel*.

Últimamente se ha llevado a cabo la combinación de los bloques lógicos e interconexiones

de los *FPGA* con microprocesadores y periféricos relacionados, con el objetivo de formar un "Sistema programable en un chip". Ejemplos de esta innovación son las tecnologías híbridas que pueden ser encontradas en los dispositivos *Kintex UltraScale* y *Virtex UltraScale* de *Xilinx* (4). Estos dispositivos incluyen uno o más procesadores *PowerPC* embebidos junto con la lógica del *FPGA*. Una alternativa a esta idea anterior es la de usar núcleos de procesadores implementados haciendo uso de la lógica del *FPGA*. Esos núcleos incluyen los procesadores *MicroBlaze* (5) y *PicoBlaze* (6) de *Xilinx*, *Nios II* (7) de *Altera*, y los procesadores de código abierto *Cortex-A* y *Cortex-A50* (8) de *ARM*.

En la actualidad existen algunas *FPGAs* que soportan la reconfiguración parcial del sistema; lo que permite que una parte del diseño pueda ser reprogramado, mientras tanto el resto pueda continuar funcionando. Esta idea es el principio de la "computación reconfigurable" o de los "sistemas reconfigurables". Ejemplo de esta característica anterior se encuentran las *FPGAs Virtex-4, Virtex-5, Virtex-6, Virtex-7, Kintex-7, Artix-7* y la *Zynq™-7000* de *Xilinx* (9); y la familia de dispositivos de la *Stratix V* de *Altera* (10).

Para sacar el máximo partido y todo el potencial que ofrecen las *FPGAs* se necesita la ayuda de un entorno de desarrollo especializado en el diseño de sistemas y de herramientas que asistan en el proceso de diseño, simulación, síntesis del resultado y configuración del hardware. *Xilinx*, en este sentido proporciona el denominado *ISE® Design Suite* (11) para todos sus dispositivos programables; disponible en tres ediciones en dependencia de la utilidad requerida.

Un diseño de usuario, sea cual sea, se puede procesar de diferentes formas; ya sea como esquemático, diagramas de estado o haciendo uso de un lenguaje de descripción de hardware (*VHDL*, *Verilog*).

De todas las *FPGAs* que existentes en la actualidad de múltiples fabricantes o empresas y que nos ofrecen prestaciones similares, se ha decidido trabajar con la *Spartan 3E-Starter Board* de *Xilinx* ya que posee varias de las características necesarias para el diseño. Esta placa presenta unas buenas características en relación prestaciones/coste.

8.2. Acerca del puerto PS/2.

El conector o puerto *PS/2* adquiere este nombre a partir de la serie de ordenadores diseñados por la compañía *IBM* llamados *IBM Personal System/2* que es creada en 1987 (12). Este estándar de conexión es empleado en los Teclados (*Keyboard*) y Ratones (*Mouse*). Esta empresa aportó muchos adelantos que fueron inmediatamente adoptados por el mercado del *PC* (*Personal Computer* u Ordenador Personal), lo cual convirtió al *PS/2* en uno de los primeros conectores. La interfaz con un teclado *PS/2* es serie y bidireccional y está controlada por microcontroladores situados en la placa base (*motherboard*).

Por otra parte, la interfaz de teclado requiere en ambos lados un colector abierto para permitir la comunicación bidireccional. El *PS/2* está siendo reemplazado por el *USB*, ya que los dispositivos *Plug and Play* (Fácil conexión y desconexión) ofrecen mayor velocidad de conexión, múltiples posibilidades de conexión de más de un periférico de forma compatible y es multiplataforma, no importa el sistema operativo donde se use, ya sea *Windows*, *MacOS* o *Linux*.

8.3. Acerca del teclado.

El inicio del *teclado* (periférico o dispositivo de entrada) se remonta a la época de 1867 cuando *Cristopher Sholes* inventó la primera máquina de escribir comercial (13). Al desarrollarse las máquinas de escribir eléctricas, y posteriormente el surgimiento de las computadoras, el teclado adoptó su diseño a partir del mecanismo que facilitaba la escritura en las máquinas de escribir según la frecuencia en la que las letras aparecían en un texto, es decir el teclado *QWERTY* (14).

Para septiembre de 1975, *IBM* dio a conocer la *IBM 5100 PC*, sin embargo ésta no incluía un teclado. Fue hasta agosto de 1981, cuando la misma compañía lanzó al mercado la *IBM 5150 PC*, la cual integraba componentes de diversos fabricantes, incluyendo así al dispositivo (15). Durante esa época, el fabricante desarrolló tres tipos con diversas teclas. Fue hasta 1984 cuando introdujo el *IBM Personal Computer/AT*, el cual contaba con un teclado que tenía 101 teclas y utilizaba una interfaz que distribuía las teclas en cuatro grupos (16), dicha disposición se convirtió en un estándar desde entonces [véase Figura 1.]:

Figura 1. Distribución de Teclas.[1]

1- Bloque de Funciones: Va desde la tecla *F1* a *F12* incluyendo el *Esc*, en tres bloques de cuatro teclas: de *F1* a *F4*, de *F5* a *F8* y de *F9* a *F12*. Funcionan de acuerdo al programa que se esté ejecutando. Por ejemplo, en muchos programas al presionar la tecla *F1* se accede a la ayuda asociada a ese programa.

2- Bloque Alfanumérico: Está ubicado en la parte inferior del bloque de funciones, contiene los números arábigos del *1* al *0* y el alfabeto organizado como en una máquina de escribir, además de algunas teclas especiales (ej. *Ç*, *@*, *&*, *$*).

3- Bloque Especial: Está ubicado a la derecha del bloque alfanumérico, contiene algunas teclas especiales como *Imp Pant*, *Bloq Despl*, *Pausa*, *Inicio*, *Fin*, *Insertar*, *Suprimir*, *RePág*, *AvPág*, y las *Flechas Direccionales* que permiten mover el punto de inserción o cursor en las cuatro direcciones.

4- Bloque Numérico: Está ubicado a la derecha del bloque especial, se activa al presionar la tecla *Bloq Num*, contiene los números arábigos organizados como en una calculadora con el fin de facilitar la digitación de cifras. Además contiene los signos de las cuatro operaciones básicas: *suma +*, *resta -*, *multiplicación ** y *división /*; también contiene una tecla de *Intro* o *Enter*. Es importante señalar que si no está activo el teclado numérico las teclas que funcionan son las 4 operaciones matemáticas (*/*, ***, *-*, *+*), la tecla *Enter* y las teclas *2*, *4*, *6* y *8* que de esta manera funcionan como las flechas direccionales.

Además hizo uso de la tecnología de *LEDS* en éste, cuyo objetivo es mostrar físicamente cuando se activan 3 teclas específicas (*Bloq Num*, *Bloq Mayús*, *Bloq Despl*). Existen diversas formas y tipos de teclado; varían en el idioma, color, funciones e incluso son inalámbricos.

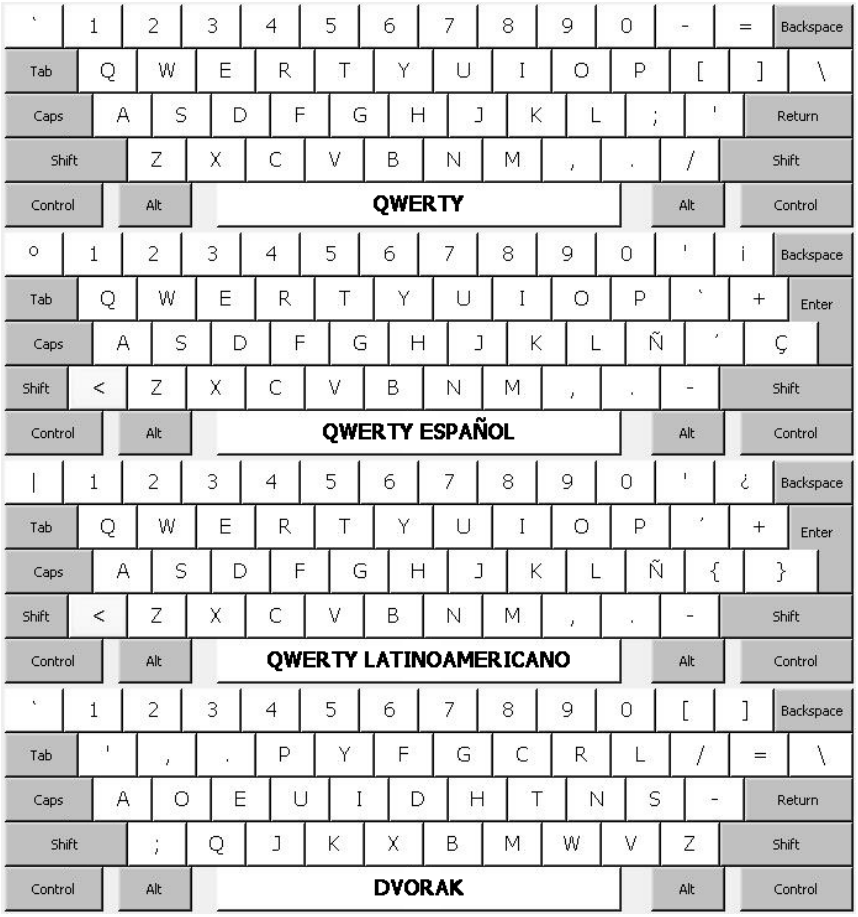
[1] Imagen tomada de Wikimedia Commons (http://commons.wikimedia.org/wiki/File:Qwerty_hispanoam%C3%A9rica.svg) y modificada.

8.3.1. Disposición de las teclas.

La distribución de teclado más usada en la actualidad es la *QWERTY* [véase Figura 2. a)]. En los teclados en su versión para el idioma Castellano existe la disposición *QWERTY Español* [véase Figura 2. b)] que además de la *Ñ*, se les añadieron los caracteres de acento agudo (´), grave (`), la diéresis (¨) y circunflejo (^), además de la cedilla (*Ç*) aunque estos caracteres son de mayor uso en francés, portugués o en catalán.

Existe además el teclado *QWERTY Latinoamericano* [véase Figura 2. c)]; que hablando estrictamente de diferencias entre estos, la forma más fácil de distinguirlos es mirando la letra Q donde el latinoamericano tiene la (*@*) como subíndice (que se activa con *Alt Gr*) y el español tiene la arroba en el número (*2*). En 1932 aproximadamente, el *Dr. Dvorak* diseño la distribución de teclado *DVORAK* [véase Figura 2. d)]. En ese teclado las vocales están en el centro a la izquierda y las consonantes más usadas a la derecha; esto hace que la escritura en ese teclado sea más simple y rápida (17).

Sobre la distribución de los caracteres en el teclado surgieron dos variantes principales: la francesa *AZERTY* [véase Figura 2. e)] y la alemana *QWERTZ* [véase Figura 2. f)]. Ambas se basaban en cambios en la disposición según las teclas más frecuentemente usadas en cada idioma. *COLEMAK* [véase Figura 2. g)] es una distribución de teclado alternativa a las distribuciones *QWERTY* y *DVORAK*, que fue desarrollada por *Shai Coleman* principalmente para el idioma inglés en enero de 2006 (18).

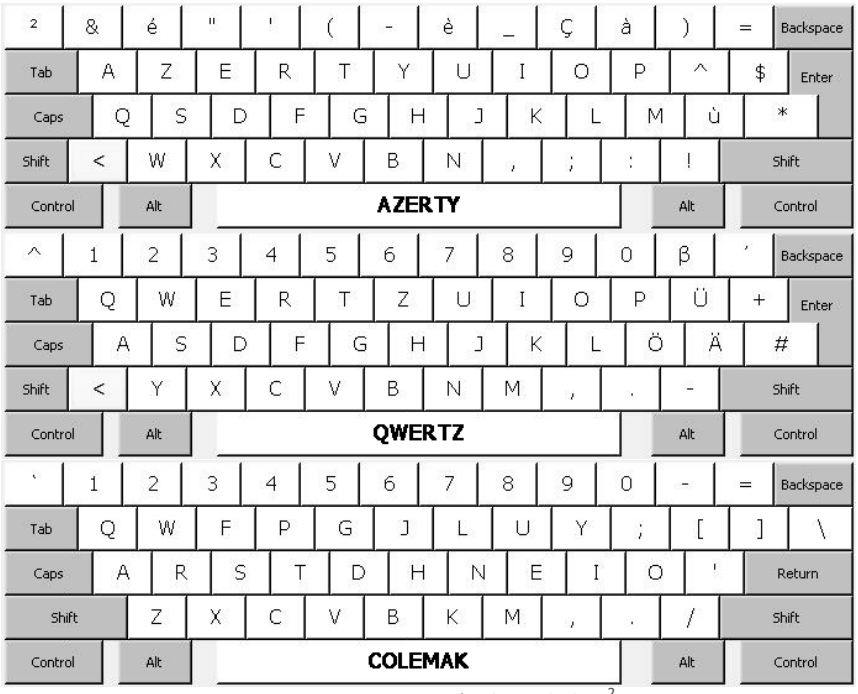


a)

b)

c)

d)



e)

f)

g)

Figura 2. Disposición de Teclados.[2]

8.3.2. Tipos de teclado.

Existen muchos modelos de teclados, en dependencia del idioma, fabricante, etc. El teclado de los *PCs* ha ido cambiando con los años, en un principio estos cambios se derivaron del lanzamiento de nuevas versiones de los *IBM PCs*. La compañía *IBM* ha desarrollado tres tipos de arquitecturas de Ordenadores Personales: el *PC/XT*, el *PC/AT* y el *PS/2* (19).

El primero (1981) de éstos tenía 83 teclas y se conectaba a la unidad mediante un cable en espiral de dos metros, lo que permitía a los usuarios cambiar su posición sin mover el resto del equipo. También incluía un teclado numérico y 10 teclas especiales (15). Más tarde (1984) apareció el teclado con 101 teclas; que consistía en un teclado alfanumérico básico, una fila de teclas de función, un teclado tipo cursor y un teclado numérico (16). Luego en 1987 *se* introduce el *IBM Personal System/2*, esta arquitectura mantiene la misma interfaz que la *PC/AT* e incorpora nuevas características. Se reduce el tamaño de los teclados debido a que se recolocan el teclado alfanumérico y el de tipo cursor, para ahorrar espacio. Además, suman tres teclas en la fila de la barra espaciadora para ajustes del propio sistema operativo. De este tipo existen tres versiones, la americana con 101 teclas como ya se ha explicado anteriormente, la europea con 102 y la asiática que basan su teclado en 106 teclas (16).

Estos teclados están quedando en desuso en la actualidad debido a la gran presencia en el mercado de los teclados *USB* y los inalámbricos. Hoy en día los sistemas operativos para los *PCs* vienen con los llamados teclados en pantalla o también llamados teclados virtuales, que son, como su propio nombre indica teclados representados en la pantalla. Estos se utilizan con el ratón o con un dispositivo especial, podría ser un *joystick* o en el caso de usar un ordenador con tecnología táctil directamente interactuar con la pantalla. Estos teclados lo utilizan personas con discapacidades que les impiden utilizar adecuadamente un teclado físico. Actualmente usar el término *AT* o *PS/2* sólo se refiere al conector porque hay una gran diversidad de ellos.

La norma internacional *ISO/IEC 9995* especifica las diversas características de los teclados.

[2] Imágenes generadas mediante el programa Microsoft Keyboard Layout Creator v1.4.

Se define un marco para el diseño de los teclados, donde un teclado se divide lógicamente en grupos y niveles, y físicamente en secciones y zonas (20). Muchos estándares nacionales de teclado definen diseños de acuerdo a la norma *ISO/IEC 9995*, aunque algunos diseños están fuera de la norma *ISO/IEC 9995*. Ejemplos de esto son estándares de facto como el teclado de programación polaco o el teclado *Microsoft* EE.UU. International (21).

8.4. Acerca del puerto VGA.

El puerto o conector *VGA* (*Video Graphics Array* o Matriz Gráfica de Video) se utiliza en la actualidad tanto para una pantalla de computador analógico estándar o *TFT*. El *VGA* es un subsistema de vídeo que se proporcionó en 1988 por *IBM* como estándar en las tarjetas de sistema de los *IBM Personal System/2* de los modelos 50XX y superiores. Este subsistema fue diseñado para cumplir los objetivos fijados para estos nuevos sistemas y para dar soporte a la compatibilidad con antiguos productos de *IBM*; mientras que al mismo tiempo proporciona un mayor rendimiento y un aumento de funcionalidad. El *IBM EGA* (*Enhanced Graphics Adapter* o Adaptador Gráfico Mejorado) fue elegido como la base de compatibilidad para la VGA, desde que *EGA* se había convertido en el estándar de vídeo para sistemas informáticos compatibles con *IBM*. Seis nuevas modalidades de funcionamiento fueron diseñadas para satisfacer las necesidades de nuevas aplicaciones empresariales y de consumo y para mejorar la ergonomía de los sistemas (22).

Conseguir un mayor rendimiento de video presenta varios problemas de diseño, dentro de los cuales se incluyen las interferencias electromagnéticas, el tamaño físico del diseño y su coste. Estos problemas se tuvieron en cuenta para la implementación de la funcionalidad del *VGA* utilizando una sola matriz de puertas y mediante el uso de una interfaz de pantalla analógica. El uso de un *DAC* de vídeo (*Digital to Analog Converter* o Convertidor Digital - Analógico) permite que el subsistema *VGA* pueda mostrar cualquier color que pertenezca a una gama de colores de 256K cuando se utiliza una pantalla a color, o 64 tonos de gris cuando se utiliza una pantalla monocromática. El subsistema *VGA* fue diseñado para proporcionar una interfaz uniforme para el color y el blanco y negro, dicha interfaz permite seleccionar el modo de color cuando se utiliza una pantalla monocromática, o el modo de blanco y negro utilizando una pantalla a color. Al *VGA* se sumó un algoritmo de color que fue diseñado e implementado en el *BIOS* (*Basic Input/Output System* o Sistema de Entrada/Salida Básico), software que permitía que los colores se mostraran en tonos de grises en una pantalla monocromática (22).

9. Bloque 1: Interfaz y protocolo de comunicación del teclado PS/2.

9.1. Descripción del problema.

En este capítulo se describe el funcionamiento del puerto *PS/2*, el funcionamiento de un teclado *PS/2* convencional alimentado a 5 V (tanto la interfaz como el protocolo de comunicaciones) y, finalmente, las características para la implementación en *FPGA*, pines y frecuencia del reloj.

9.2. Puerto PS/2.

La interfaz física del puerto *PS/2* tiene dos tipos de conectores: el conector *DIN 5* o el conector *6-Pin Mini-DIN*. Ambos conectores son eléctricamente similares, en la práctica la diferencia está en la disposición de los pines y en su número. El conector *6-Pin Mini-DIN* tiene la distribución de pines mostrada en la Figura 3 (23).

6-Pin Mini-DIN (PS/2)	Señal
1	Datos (PS2D)
2	No Implementado
3	Tierra (GND)
4	Alimentación (V_{CC} = +5V)
5	Reloj (PS2C)
6	No Implementado

Tabla 1. Distribución de pines del puerto PS/2 (24).

Figura 3. Conector 6-Pin Mini–DIN (24).

Los pines que interesan y que se van a utilizar son el 1 y el 5, como se aprecia resaltado en la Tabla 1 y señalado en la Figura 3. A partir de este momento en el documento se utilizará el término de *host* cuando se quiera hablar de la *FPGA* y el término de *dispositivo* cuando se quiere hacer referencia al teclado.

En lo que respecta a la interfaz eléctrica del puerto *PS/2* se puede decir que los pines *V_{CC}/GND* proporcionan la energía al dispositivo. El teclado no puede soportar más de 275 mA proporcionados en este caso por la *FPGA*, con lo cual se debe tener cuidado para evitar sobretensiones transitorias. Esto puede ocurrir cuando la *FPGA* está encendida y se conecta/desconecta el teclado, por este motivo se recomienda que primero se conecte el dispositivo y luego se alimente el host. En general el teclado tiene unas especificaciones de potencia que son las siguientes, V_{CC} puede variar entre 4.5 V y 5.5 V y la corriente máxima que soporta es de 275 mA (23).

Las líneas de datos y de reloj, ambas son de *colector abierto* con resistencias de *pull-up* a V_{CC}. La interfaz de colector abierto tiene dos estados posibles: *nivel bajo* o estado de *alta impedancia*. En el estado de nivel bajo, un transistor pone la línea a nivel bajo '0'; y en el estado de alta impedancia, la interfaz actúa como circuito abierto, por lo que la línea no conduce 'Z'. Por otra parte, una resistencia de *pull-up* se conecta entre el bus y V_{CC} por lo que el bus se pone a nivel alto '1' si ninguno de los dispositivos en el bus están poniéndolos a nivel bajo '0'. El valor de esta resistencia no es demasiado importante pero varía entre 1 y 10 kΩ. Esta interfaz descrita anteriormente se observa en la Figura 4 (23).

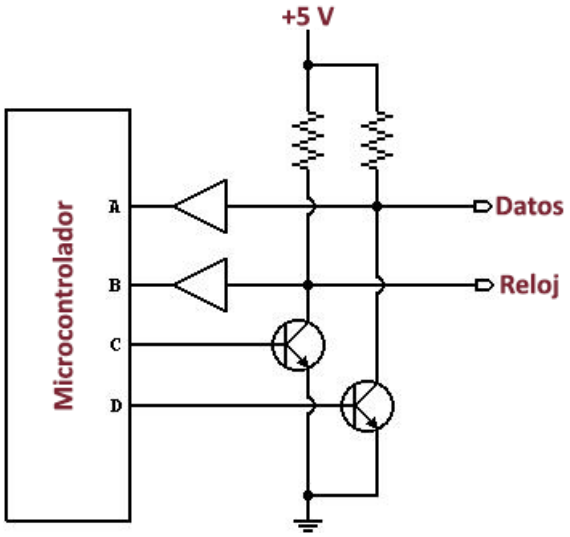


Figura 4. Interfaz general de colector abierto.

Los datos y el reloj se leen en los pines del microcontrolador A y B, respectivamente. Ambas líneas se mantienen normalmente a +5 V, pero se pueden poner a tierra al imponer la lógica '1' en C y D. En consecuencia, la línea de datos es igual a D invertida; y la línea de reloj es igual a C invertida (23).

9.3. Teclado PS/2.

El puerto *PS/2* del teclado implementa un protocolo serie síncrono bidireccional. El bus está *inactivo* cuando las dos líneas están a nivel alto (colector abierto). Este es el único estado donde el dispositivo puede comenzar a transmitir datos. El host tiene el último control sobre el bus y puede inhibir la comunicación en cualquier momento, poniendo la línea de reloj a nivel bajo '0'.

El teclado siempre genera la señal de reloj. Si el host desea enviar datos, primero debe inhibir la comunicación desde el dispositivo poniendo la línea de reloj a nivel bajo. Entonces el host pone la línea de datos a nivel bajo y emite la señal de reloj. Este es el estado "*Request to Send*" y señala al dispositivo para comenzar a generar los impulsos de reloj [ver Tabla 2.] (23).

Línea de Datos	Línea de Reloj	Especificación
Alto o '1'	Alto o '1'	El dispositivo está en estado inactivo, donde está listo para enviar información al host. Este es el estado que interesa en este trabajo.
Alto o '1'	Bajo o '0'	Comunicación inhibida, estado en que el host interrumpe la comunicación debido a algún fallo.
Bajo o '0'	Alto o '1'	"Request to Send" o solicitud de envío del host, estado en que el host está listo para enviar información al dispositivo.

Tabla 2. Estados del bus de comunicaciones (23).

9.3.1. Protocolo.

Todos los datos se transmiten en un byte a la vez y cada byte se envía en un marco de 11 a 12 bits; en este proyecto el dispositivo envía información al host solo con 11 bits. Estos bits son los siguientes:

- **Bit 1:** 1 bit de arranque o comienzo, este siempre es '0'.
- **Bits del 2 al 9:** 8 bits de datos, el bit menos significativo siempre va primero.
- **Bit 10:** 1 bit de paridad, que en este protocolo es de paridad impar.
- **Bit 11:** 1 bit de parada o fin, este siempre es '1'.

El bit de paridad se establece a '1' si existe un número par de *unos* en los bits de datos y se pone a '0' si hay un número impar de *unos* en los bits de datos. El número de *unos* en los bits de datos más el bit de paridad siempre suman un número impar (paridad impar). Esto se utiliza para la detección de errores; el teclado debe comprobar este bit y si es incorrecto se debe responder como si hubiera recibido un comando no válido.

Los datos enviados desde el dispositivo al host se leen en el flanco de bajada de la señal de reloj y los datos enviados desde el host al dispositivo se leen en un flanco de subida, como se puede apreciar en la Figura 5 (23).

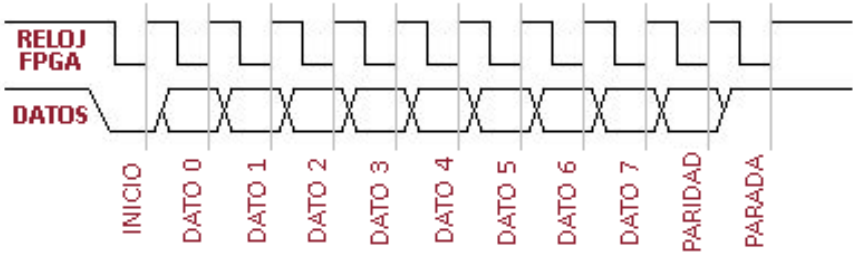


Figura 5. Comunicación del dispositivo al host.
La línea de datos cambia de estado cuando el reloj está a nivel alto y estos datos son válidos cuando el reloj está a nivel bajo (23).

La frecuencia de reloj debe estar en el rango de 10 a 16.7 kHz; lo que significa que el reloj debe estar a nivel alto de 30 a 50 µs y a nivel bajo de 30 a 50 µs, tal como se observa en la tabla 3. Además el tiempo transcurrido desde el flanco de subida de un pulso de reloj a una transición de datos debe ser de al menos 5 µs. El tiempo transcurrido desde una transición de datos a un flanco de bajada de un pulso de reloj debe ser de al menos 5 µs y no más de 25 µs, como se puede la Tabla 3 y la Figura 6 (23).

Símbolo	Parámetro	Min	Max
T_{CK}	Tiempo de reloj a nivel alto o bajo.	30 µs	50 µs
T_{SU}	Tiempo transcurrido desde una transición de datos a un flanco de reloj.	5 µs	25 µs
T_{HLD}	Tiempo transcurrido desde un flanco de reloj a una transición de datos.	5 µs	25 µs

Tabla 3. Tiempos del bus *PS/2* (24).

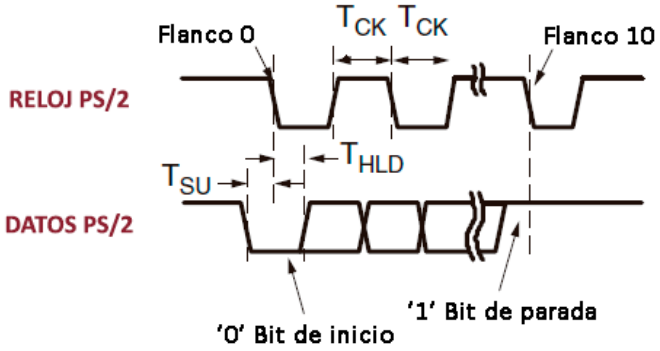


Figura 6. Tiempos de las formas de ondas del bus *PS/2* (24).

Como se explicó anteriormente las líneas de datos y de reloj son de colector abierto; una resistencia se conecta entre cada línea y +5 V, por lo que el estado de reposo del bus es a nivel alto. Cuando el teclado quiere enviar alguna información, este primero comprueba la línea de reloj para asegurarse de que está a nivel lógico alto. Si esto no fuera así, es porque el *host* está inhibiendo la comunicación y el dispositivo debe almacenar cualquier dato que quiera enviar hasta que el *host* libere el reloj nuevamente. La línea de reloj debe estar continuamente a nivel alto, al menos durante 50 µs antes de que el dispositivo pueda comenzar a transmitir sus datos. El *host* puede inhibir la comunicación en cualquier momento, poniendo la línea de reloj a nivel bajo durante al menos 100 µs (23).

9.3.2. Interfaz.

Los teclados consisten en una matriz grande de teclas, las cuales son controladas por un procesador que se encuentra dentro del propio teclado y es llamado el "codificador de teclado". El procesador específico varía de un teclado a otro, pero básicamente todos hacen lo mismo: controlar qué tecla(s) esta(n) siendo presionada(s)/liberada(s) y enviar los datos correspondientes al host. Este procesador se encarga de toda la supresión de rebotes y topes de los datos en su buffer de 16 bytes, si es necesario. En este trabajo lo que se pretende es diseñar un "controlador de teclado" o "driver" que se encargará de decodificar todos los datos recibidos desde el teclado (25).

El procesador de teclado pasa la mayor parte del tiempo explorando o controlando la matriz de teclas. Si considera que cualquier tecla se presiona, se libera o se mantiene pulsada, el dispositivo enviará un paquete de información conocida como *scancode* al *host*. Existen dos tipos diferentes de *scancodes*: *makecode* y *breakcode*. El *makecode* se envía cuando se pulsa una tecla o se mantiene presionada; cuando se presiona y mantiene presionada una tecla, esa tecla se convierte en *typematic*, lo que significa que el teclado seguirá enviando el *makecode* de la misma hasta que la tecla se libera o se pulsa otra. En el caso de que más de una tecla se mantenga presionada, sólo la última tecla presionada se convierte en *typematic*. La repetición del *typematic* se detiene cuando se suelta esta última tecla, a pesar de que otras teclas puedan estar presionadas (25).

El *breakcode* se envía cuando se suelta una tecla. Cada tecla tiene asignado su propio y único *makecode* y *breakcode*, por lo que el host puede determinar con exactitud lo que pasó con la tecla mirando su único *scancode*. Los *makecodes* y *breakcodes* de todas las teclas se agruparon en lo que se denomina *scancode set*. Hay tres estándar *scancodes sets* llamados 1, 2 y 3, como se puede ver en la Tabla 4. Desafortunadamente para saber cuáles son los *scancodes* que corresponden a cada tecla no hay una fórmula sencilla; es decir, si se quiere saber el *makecode* y *breakcode* que corresponde a una tecla específica habrá que buscarlos en uno de estos *scancodes sets* (25).

Conjunto	Descripción
Scancode Set 1	*Scancode set* original XT, con el apoyo de algunos teclados modernos.
Scancode Set 2	Por defecto es el *scancode set* para todos los teclados modernos. Este es el que se usa en este trabajo.
Scancode Set 3	*Scancode set* opcional *PS/2*, rara vez se utiliza.

Tabla 4. Estándar scancodes sets (25).

Cada vez que se pulsa una tecla, el *makecode* de esta se envía al *host*. Es necesario tener en cuenta que este *makecode* sólo representa una tecla en un teclado, no representa el carácter impreso en esa tecla. Esto significa que no hay una relación definida entre un *makecode* y un código *ASCII*. Depende del host el traducir los *scancodes* a caracteres o comandos (25).

Aunque la mayoría de los *makecodes* del *scancode set* 2 son sólo de un byte de ancho, hay un puñado de "teclas extendidas" cuyos *makecodes* son de dos o cuatro bytes de ancho. Estos *makecodes* se pueden identificar por el hecho de que su primer byte es *E0h*, pero en este trabajo no interesan este tipo (25).

La mayoría de los *breakcodes* del *scancode set* 2 son de dos bytes de longitud, donde el primer byte es *F0h* y el segundo byte es el *makecode* de esa tecla. Los *breakcodes* de las teclas extendidas son generalmente de tres bytes de longitud, donde los dos primeros bytes son *E0h*, *F0h*, y el último byte es el último byte del *makecode* de esa tecla, como se ha dicho anteriormente en este trabajo no interesan las teclas extendidas. Como ejemplo en la Tabla 5, se han enumerado los *makecodes* y *breakcodes* de algunas teclas del *scancode set* 2 (25):

Tecla	(Set 2) *Makecode*	(Set 2) *Breakcode*
"A"	1C	F0, 1C
"5"	2E	F0, 2E
"F10"	09	F0, 09
Flecha derecha	E0, 74	E0, F0, 74
"Ctrl" Derecho	E0, 14	E0, F0, 14

Tabla 5. *Makecodes* y *breakcodes* del *scancode set* 2 (25).

Si la transmisión se inhibe antes del impulso número 11 de reloj, el dispositivo debe interrumpir la transmisión actual y preparar la retransmisión del *pedazo* de datos cuando el *host* emita el reloj otra vez. Un pedazo de los datos podría ser un *makecode* o un *breakcode*. Por ejemplo, si un teclado se interrumpe durante el envío del segundo byte de 2 bytes de *breakcode*, será necesario que vuelva a retransmitir ambos bytes del *breakcode*, no sólo el byte que se interrumpió. Si el *host* pone a nivel bajo el reloj antes de la primera transición de reloj de nivel alto a nivel bajo, o después del flanco de bajada del último impulso de reloj, el teclado no necesita retransmitir ningún dato. Sin embargo, si se crean nuevos datos que deben ser transmitidos, estos tendrán que ser almacenados hasta que el *host* emita el reloj de nuevo. Los teclados tienen un búfer de 16 bytes para este fin. Si se producen más de 16 bytes válidos de las pulsaciones de teclado, las pulsaciones de teclado adicionales serán ignoradas hasta que haya espacio en el búfer (23).

Para entender lo explicado anteriormente se pondrá un ejemplo. Si se envía el carácter 'q' la secuencia de *makecodes* y *breakcodes* que se transmite a la *FPGA* sería la siguiente. Como se trata de una letra en minúscula, el orden de acontecimientos que debe tener lugar es: pulsar la tecla "q" y soltar la tecla "q". Los *scancodes* asociados a estos eventos son los siguientes: *makecode* de la tecla "q" (15h) y *breakcode* de la tecla "q" (*F0h*, *15h*). Por lo tanto, los datos enviados al host serían: <15h>, <F0h> y <15h>. En la Figura 7 se muestra la salida del puerto *PS/2* vista en un osciloscopio cuando se envía el *makecode* de la tecla "q" (23).

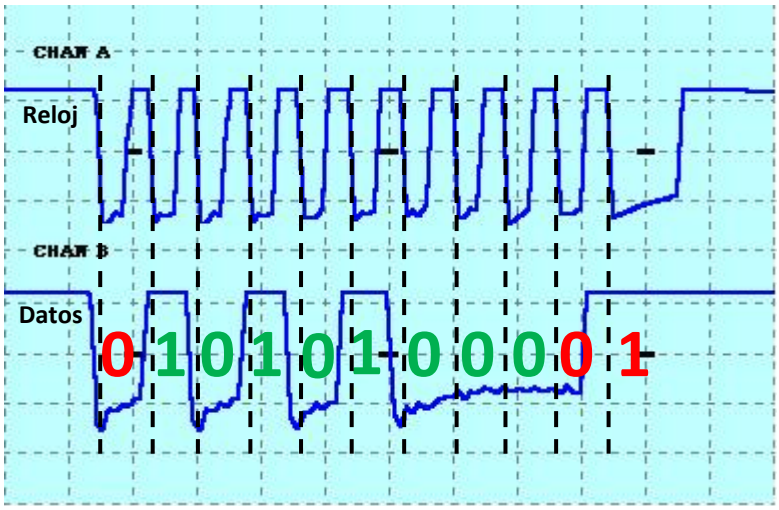


Figura 7. *Scancode* de la tecla "q", <15h> = 1010 1000b (23).

9.4. Especificaciones del bloque 1.

9.4.1. Interfaz.

En este apartado se especifica cuáles son las entradas y salidas (E/S) de este bloque 1. Estas se pueden apreciar en el anexo de esta memoria, exactamente en subtitulo 16.1 más adelante. Las señales de este bloque son:

Entradas

- **ps2_clk**: es la señal del reloj que produce el teclado, la cual llega a la *FPGA* por el puerto *PS/2* y se modifica para adaptarse al reloj del *FPGA*. La frecuencia debe estar en el rango de 10 a 16.7 kHz como se especifica en el apartado 9.3.3.
- **ps2_dato**: es la señal que lleva a la *FPGA* los datos de cada tecla presionada o liberada a través del puerto *PS/2*. La actividad de esta señal está documentada en el apartado 9.3.
- **clk**: es la señal que representa el reloj de la *FPGA*, el cual fija todo el diseño. Tiene una frecuencia de 50 MHz lo que produce un periodo de 20 ns.
- **rst**: señal de inicialización asíncrona del sistema, activa a nivel alto.

Salidas

- **error**: señal activa a nivel alto que indica la presencia de algún fallo por cualquier razón.
- **tecla**: señal de 8 bits que representa el código en binario de la tecla presionada o liberada del teclado.

9.4.2. Funcionalidad.

Se pretenden crear un diseño que sea independiente de la tecnología, es decir, que se pueda usar en un futuro con cualquier dispositivo. Para el desarrollo de este trabajo se utilizó la placa *Spartan 3E-Starter Board* que dispone entre otras características de un puerto *PS/2* para ratón o teclado, un puerto *VGA* para pantallas, 8 *LEDs* y un conjunto de interruptores. Dentro de las características de esta *FPGA* [véase Figura 8.] la más importante y la que más se tendrá en cuenta para el desarrollo del controlador o driver de teclado es su *reloj principal*; cuyo valor de frecuencia es de 50 MHz, lo que implica que tenga un periodo de 20 ns. Este reloj tiene un pin asignado en la placa, el cual se muestra en la Tabla 6.

En este primer bloque la idea es captar el código binario de cada tecla presionada o liberada en el teclado, identificar de qué tecla se trata y luego guardarla en un registro para saber exactamente con que tecla se está trabajando. Para mostrar este código binario almacenado en el registro se utilizarán los 8 *LEDs* de la placa, cuyos pines también son fijos y se muestran en la Tabla 6. Además todo este diseño se basa en el protocolo *PS/2* que utiliza dos pines más que son el ps2_clk y el ps2_dato cuyas direcciones también se muestran en la Tabla 6. Por último, la mayoría de los circuitos en su diseño llevan un *Reset* para volver al estado inicial por si hubiera algún problema y una señal especifica de error como si fuera un *flag*; en este trabajo se asignó ambas entradas a dos pines específicos de la *FPGA*, cuyas direcciones vienen en la Tabla 6.

Elemento	Pin
Reloj	C9
Reset	L13
Error	A4
Reloj PS/2	G14
Dato PS/2	G13
LED 1	F12
LED 2	E12
LED 3	E11
LED 4	F11
LED 5	C11
LED 6	D11
LED 7	E9
LED 8	F9

Tabla 6. Pines de la *FPGA* para el bloque 1 (24).

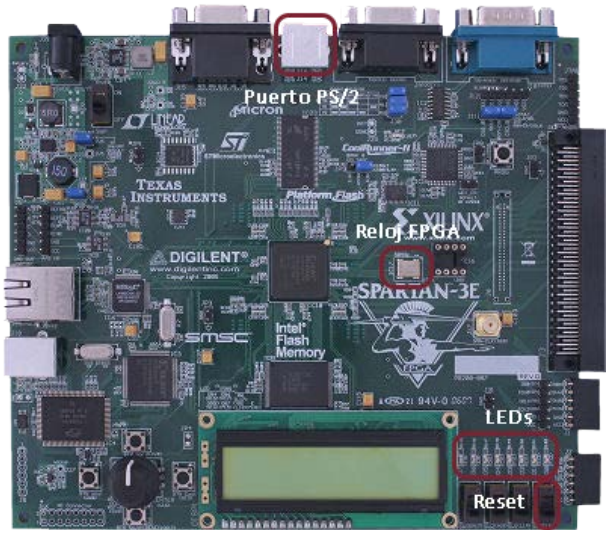


Figura 8. Características usadas de la placa *Spartan 3E-Starter Board* para el bloque 1 (24).

9.4.3. Arquitectura.

En esta sección sólo se detallan las partes o bloques del circuito, que se explicará con detalle en el apartado 9.4.4 más adelante. En la Tabla 7 se enumeran todos los bloques del circuito y en la Figura 9 se puede ver el diagrama completo de la arquitectura.

Bloque	Descripción
1	Detector de flancos de bajada.
2	Detector de flancos de subida.
3	Detector de flancos.
4	Registro de desplazamiento de 10 bits.
5	Registro auxiliar de 8 bits.
6	Registro de salida de 8 bits.
7	Temporizador de 50 µs.
8	Detector de fin de la cuenta del temporizador.
9	Detector de nivel '1' y prolongado del reloj *PS/2*.
10	Detector de error.
11	Biestable para actualizar la señal de error.
12	Contador de 11 flancos de bajada.
13	Detector de *breakcode*.
14	Detector de *scancode*.
15	Máquina de estados.

Tabla 7. Resumen de los bloques del bloque 1.

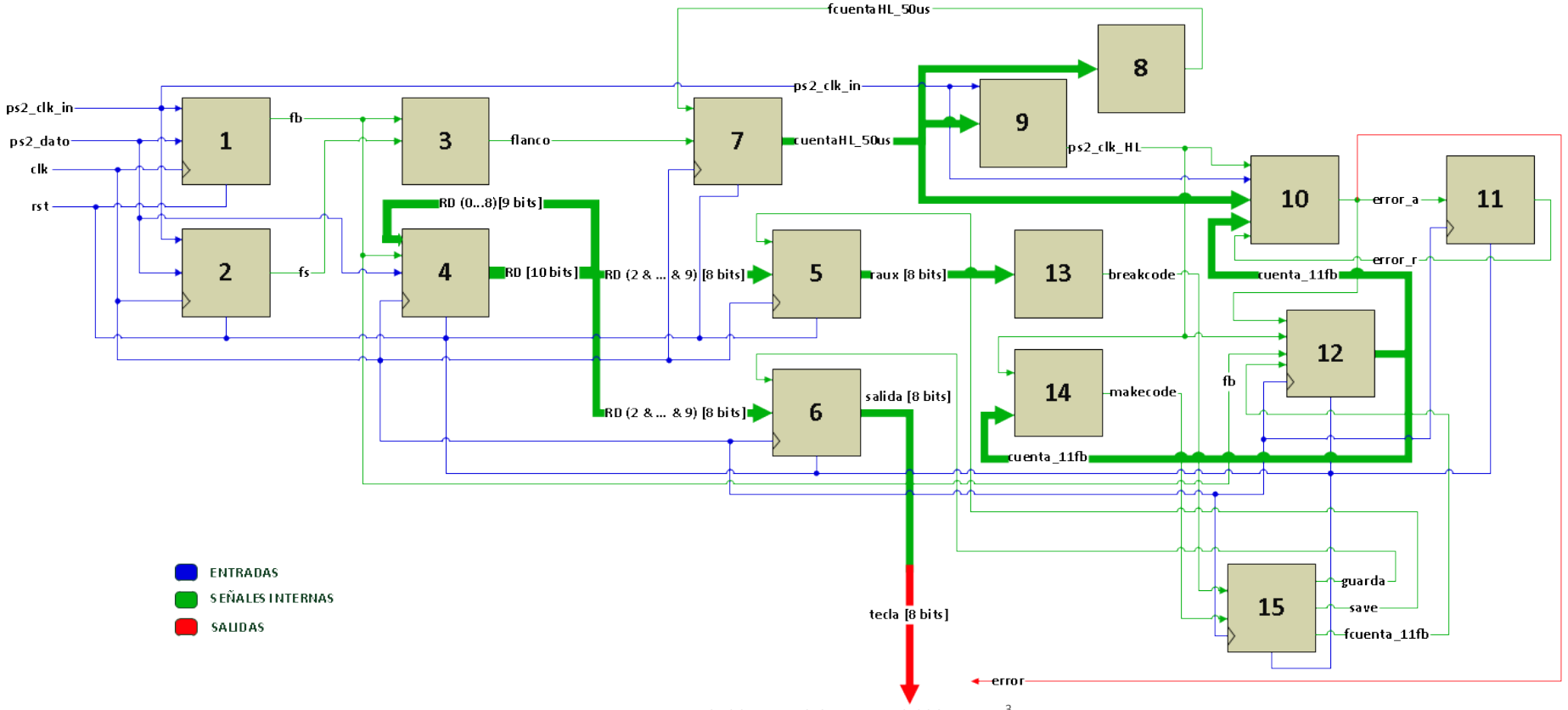


Figura 9. Diagrama de bloques del circuito del bloque 1.[3]

[3] En este diagrama de bloques se han numerado las cajas para facilitar la explicación y detalle a posteriori, aunque se sabe que deberían tener sus propios nombres.

9.4.4. Diseño detallado y síntesis del circuito.

Como se mencionó en el apartado 9.4.3, ahora se explicará con detalle cada parte del circuito diseñado. Antes de proceder a la explicación es necesario aclarar que sólo se ha detallado el diseño de los detectores de flancos, debido a que fueron los más interesantes a la hora de su creación. Se sabe que no son diseños complicados y que no llevan mucho estudio de fondo, pero si tienen un nivel de dificultad más alto comparado con contadores, registros y/o blanqueadores.

El detector de flancos de bajada (bloque 1) es el encargado de saber cuándo en la línea del *ps2_clk_in* hay un flanco de bajada. Es necesario saber cuándo cambia esta línea de nivel alto a bajo, ya que de esta forma se sabe cuándo el teclado comienza a transmitir algo o comprobar el tiempo de los diferentes semiciclos a nivel bajo de la transmisión. Para ello se ha desarrollado una lógica combinacional formada por dos biestables tipo *'D'*, una puerta *NOT* y otra *NOR* [ver Figura 10.]. Básicamente su función es detectar cuando la entrada *ps2_clk_in* (en este caso son las señales *DB1* y $\overline{DB0}$) se pone a nivel bajo o *'0'* durante dos ciclos seguidos de la señal de reloj *clk* [ver Tabla 8.].

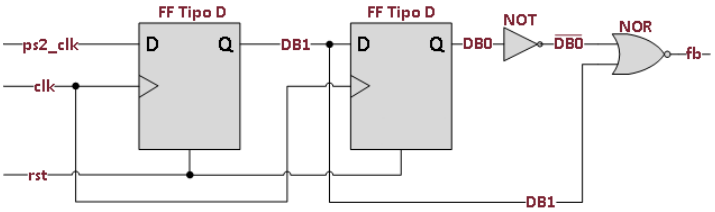


Figura 10. Lógica combinacional del detector de flancos de bajada.

DB1	DB0	$\overline{DB0}$	DB1 **NOR** $\overline{DB0}$
0	0	1	0
0	1	0	1
1	0	1	0
1	1	0	0

Tabla 8. Tabla de verdad del FF_DFB.

El detector de flancos de subida (bloque 2) se encarga de encontrar cuando cambia la línea *ps2_clk_in* de nivel bajo a alto, esto sirve para temporizar los semiciclos positivos y saber cuándo se acaba una transmisión de datos del dispositivo. Este segundo bloque se diseñó usando también una lógica combinacional formada por dos biestables tipo *'D'*, una puerta *NOT* y otra puerta *AND* [ver Figura 11.]. La función de esta lógica combinacional es detectar cuando la entrada *ps2_clk*_in (en este caso *DS1* y $\overline{DS0}$) se pone a nivel alto o *'1'* durante dos ciclos seguidos de la señal de reloj *clk* [ver Tabla 9.]. Derivado de ambos detectores surge el detector de flancos, que es el encargado de informar cuando hay un *flanco*, tanto de bajada o de subida (bloque 3).

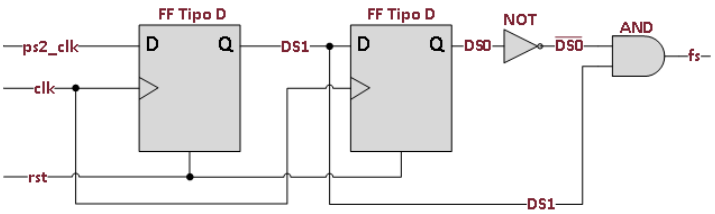


Figura 11. Lógica combinacional del detector de flancos de subida.

DS1	DS0	$\overline{DS0}$	DS1 **NOR** $\overline{DS0}$
0	0	1	0
0	1	0	0
1	0	1	1
1	1	0	0

Tabla 9. Tabla de verdad del FF_DFS.

Se utiliza un registro de desplazamiento de 10 bits (bloque 4) para ir almacenando los datos recibidos del teclado en cada flanco de bajada. En un primer diseño se utilizaba un registro de 11 bits para almacenar la trama de 11 bits completa, pero cuando se sintetizaba en el *ISE* salía un *warning* que informaba que el último bit de ese registro no se utilizaba; por lo

que se eliminó. Más adelante en la sección 9.4.5 de simulaciones se detalla el funcionamiento de este registro y se aprecia como con solo 10 bits funciona a la perfección [ver Figura 16].

El bloque 5 es un registro de 8 bits cuyo objetivo es almacenar cada *scancode* recibido, ya sea *makecode* o *breakcode*, y poder tener esa información disponible hasta el siguiente envío. Es decir, ese registro es el que permite que la máquina de estados sepa que *scancode* se recibió del teclado, mientras que se sigue recibiendo en el registro de desplazamiento la información enviada por el *ps2_dato*. Este registro concatena los bits del segundo al noveno del registro de desplazamiento y guarda cada *scancode* cuando se activa la señal que viene de la máquina de estados *save.* El bloque 6 es un registro igual exactamente que el del bloque 5 pero es para guardar solamente el *makecode* de la tecla presionada, este registro almacena dicho código cuando se activa la señal *guarda* después de recibir un *breakcode*.

El temporizador de 50 µs (bloque 7) cuenta el tiempo tanto del semiciclo positivo como el negativo, los cuales tienen que cumplir con el protocolo de comunicaciones del puerto *PS/2* entre 30 y 50 µs cada semiciclo. Por este motivo, para poder detectar cuando la línea del *ps2_clk_in* se queda a nivel alto o bajo más de ese tiempo, el temporizador está diseñado para que cuente un ciclo más. Este temporizador se reinicia con cada flanco detectado y además cuando la cuenta llegue a su fin, en este caso a 2500 ciclos; esto es lo que hace el bloque 8 activar la señal *fcuentaHL_50us* cuando pasen los 2500 ciclos.

El bloque 9 tiene la función de activar la señal *ps2_clk_HL* cuando la entrada *ps2_clk_in* se encuentra a nivel alto y el temporizador ha finalizado la cuenta de los 2500 ciclos. Esto indica que el reloj del teclado está inactivo y se mantiene a nivel alto, es decir, no hay ninguna transmisión actual.

Se diseñó un detector de errores (bloque 10), este activa la señal *error_a* cuando el temporizador ha terminado su cuenta y la línea de reloj del teclado está a nivel bajo, o cuando la línea de reloj del teclado está inactiva y no han pasado los 11 bits de información, o sea, que se interrumpió la transmisión antes que acabara. Seguido de este bloque se encuentra un biestable (bloque 11) para actualizar la señal de *error_r,* la cual cambia con cada ciclo de reloj de la *FPGA* dependiendo del valor que tenga la señal *error_a*. Además de mantener el valor de error del ciclo anterior, si hubiese algún *Reset* es que el borra el valor de la salida *error.*

El bloque 12 es un contador, el cual cuenta los 11 flancos de bajada para saber que ha pasado una trama completa. El contador se reinicia cuando se activa la señal *fcuenta_11fb* que proviene de la máquina de estados o cuando se activa la señal *ps2_clk_HL* (está inactiva la línea de reloj del teclado) o cuando ha ocurrido algún error.

La entrada del bloque 13 es la salida del registro de 8 bits auxiliar *raux*, el que guarda todos los *makecodes*. La salida del bloque 13 *breakcode* es una señal que se activa solo cuando este *scancode* es el *breakcode* (F0 h = 1111 0000 b), de lo contrario se mantiene a '0'. Esta salida es una de las entradas de la máquina de estados, es una de las condiciones para ir a un estado u otro.

La salida del bloque 14 *makecode* solo se activa cuando se han recibido 11 bits (una trama completa) y la línea de reloj del teclado esta inactiva, de lo contrario se mantiene a '0'. Esta es la otra señal de entrada de la máquina de estados, que es otra de las condiciones que verifica para decidir a qué estado continúa.

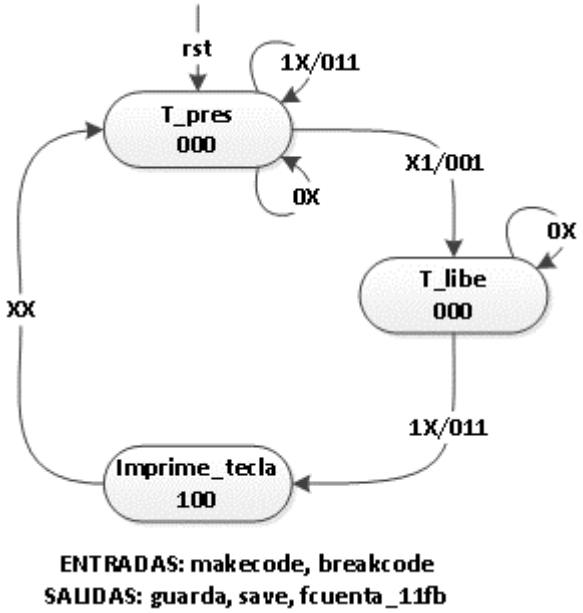


Figura 12. Máquina de estados del bloque 1.

Por último está la mencionada anteriormente máquina de estados (bloque 15), esta es muy simple y solo está formada por 3 estados, dos entradas que conforman las posibles transiciones y 3 salidas. En la Figura 12 se puede ver su diseño al detalle. Si se observa esta imagen se aprecia como después de un *Reset* manual se posiciona en el primer estado que se llama *T_pres* donde las 3 salidas están a '0', se mantendrá en ese estado siempre que no se haya recibido un *scancode*, si este se recibe también se mantiene en el mismo estado pero se activan las salidas *save* y *fcuenta_11fb*. Si hubiese un *breakcode* pasaría al siguiente estado llamado *T_libe* y activaría la señal *fcuenta_11fb*. En este estado también las 3 salidas están a '0'; se quedará en este estado mientras la señal *makecode* esté a '0' y en cuanto cambie pasaría al tercer y último estado llamado *Imprime_tecla*, activando las señales *save* y *fcuenta_11fb*. Una vez que se encuentre en el tercer estado solo pone la salida *guarda* a '1' y continúa en el siguiente ciclo de reloj de la *FPGA* al primer estado.

Una vez terminado el controlador o driver de teclado cumpliendo todas las especificaciones del protocolo y la interfaz de teclado *PS/2* se sintetiza todo el código en el programa de *Xilinx ISE Project Navigator*. Este software se encarga de verificar, depurar, corregir e implementar el código hecho. Cuando se analiza dicho código, se sintetiza, se implementa y se transfiere a la *FPGA*. El *Project Navigator* avisa de los posibles *warnings* o errores, además de una serie de informes informativos en los cuales nos informa de parámetros importantes del diseño actual.

Dentro de estos informes se encuentra uno muy importante que es el resumen de síntesis, donde nos informan acerca del número de unidades lógicas que ocupa el diseño del total disponible en la *FPGA*, ya sean *Flip Flops*, *LUTs* o *Slices*. Toda esta información se ofrece en cantidades y en porcentajes. En la Figura 13 se muestra el resumen del diseño del controlador o driver de teclado final y completamente funcional.

Como se aprecia en la Figura 13, el resultado obtenido del análisis es que el número de *Slices* ocupados es 52 de 4656 disponibles, lo que equivale al 1% de utilización. El número de *Flip-Flops* es 49 de 9312 lo que equivale al 1% de utilización. El número de *4 input LUTs* es 54 de 9312 que corresponde al 1% del total disponible. El número de *bonded IOBs* es 13 de 232 que es el 5%. El número de *BUFGMUXs* es 1 de 24 que es el 4% del total.

Device Utilization Summary			
Logic Utilization	**Used**	**Available**	**Utilization**
Number of Slice Flip Flops	49	9,312	1%
Number of 4 input LUTs	54	9,312	1%
Number of occupied Slices	52	4,656	1%
Number of Slices containing only related logic	52	52	100%
Number of Slices containing unrelated logic	0	52	0%
Total Number of 4 input LUTs	65	9,312	1%
Number used as logic	54		
Number used as a route-thru	11		
Number of bonded IOBs	13	232	5%
Number of BUFGMUXs	1	24	4%
Average Fanout of Non-Clock Nets	3.01		

Figura 13. Resumen de la utilización de la *FPGA* bloque 1.

Otro de los informes importantes del *Project Navigator* es el Resumen de Tiempos, el cual ofrece varios datos a tener en cuenta a la hora de utilizar este diseño en otras placas o *FPGAs*. En este caso se obtuvieron los siguientes resultados mostrados en la Tabla 10:

Minimum period: 7.000ns (Maximum Frequency: 142.857MHz)
Minimum input arrival time before clock: 6.483ns
Maximum output required time after clock: 8.161ns
Maximum combinational path delay: 7.688ns

Tabla 10. Extracto del informe de tiempos de la *FPGA* bloque 1.

En la tabla se muestra que el período mínimo de una señal es de 7 ns, lo cual implica que la frecuencia máxima es de 142,857 MHz; o sea, que si se utilizara este diseño con otra *FPGA* habría que asegurarse que esta tiene un reloj con una frecuencia igual o menor que la frecuencia calculada, porque si el período fuese menor no funcionaría el circuito. Otro dato que aparece es el tiempo mínimo que demora en llegar una señal de entrada antes de un pulso de reloj, que en este caso es 6,483 ns y el tiempo máximo que requiere una señal de salida después de un pulso de reloj es de 8,161 ns. Además informa que el retraso máximo en una ruta combinacional es de 7,688 ns. Toda esta información es crucial a la hora de usar este diseño en otras *FPGAs*. Es imprescindible cumplir con los tiempos y las frecuencias especificadas anteriormente para asegurar un funcionamiento correcto.

9.4.5. Descripción del banco de pruebas del bloque 1.

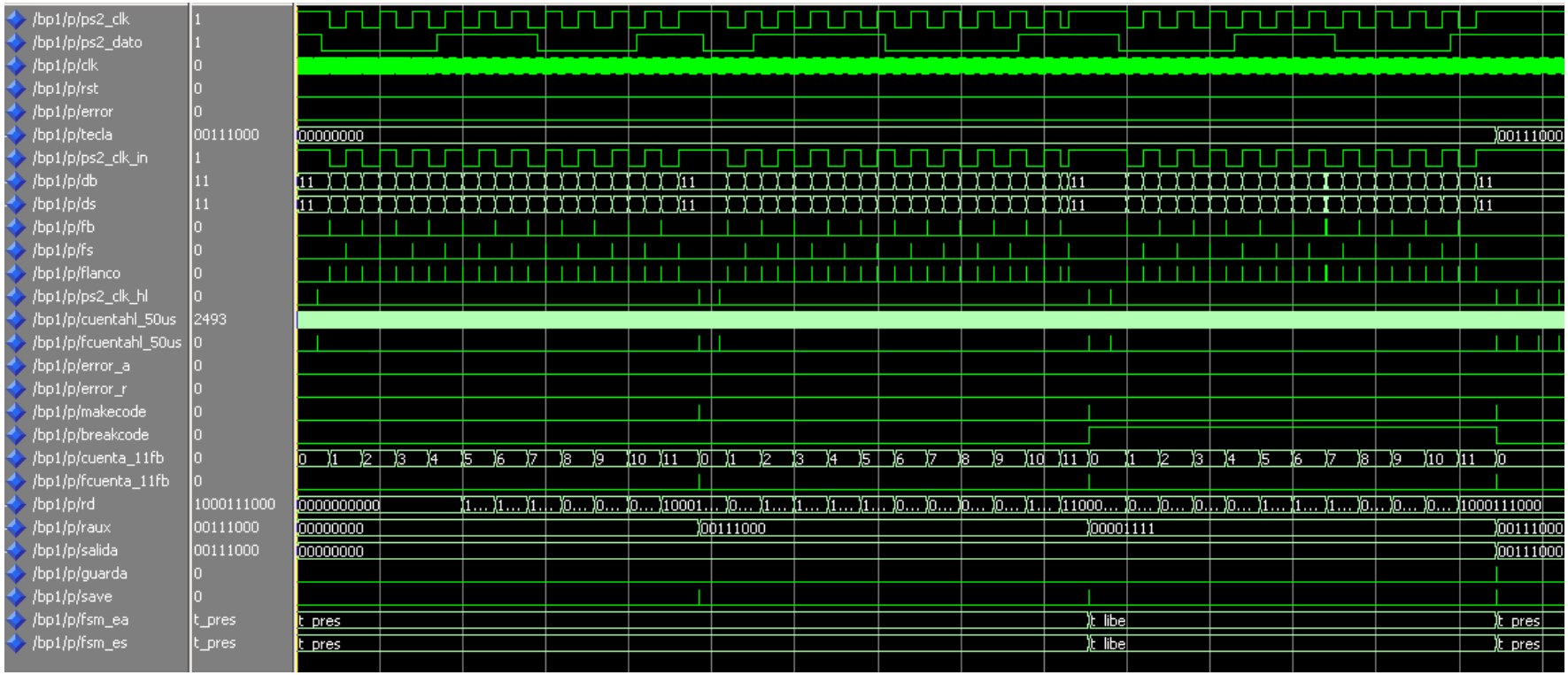


Figura 14. Simulación 1.

En esta simulación se aprecia el envío de la tecla o carácter *'A'* = "1C h = 0001 1100 b". La secuencia de sucesos es: 1- *makecode* de la tecla *'A'* <1C> 2- *breakcode* de la misma tecla <F0><1C>. En efecto se verifican estos acontecimientos y además se ve que la señal *makecode* se activa 3 veces, al igual que la señal *save*; pero sólo se registra en la salida *tecla* el último byte del *breakcode* cuando se activa la señal *guarda*.

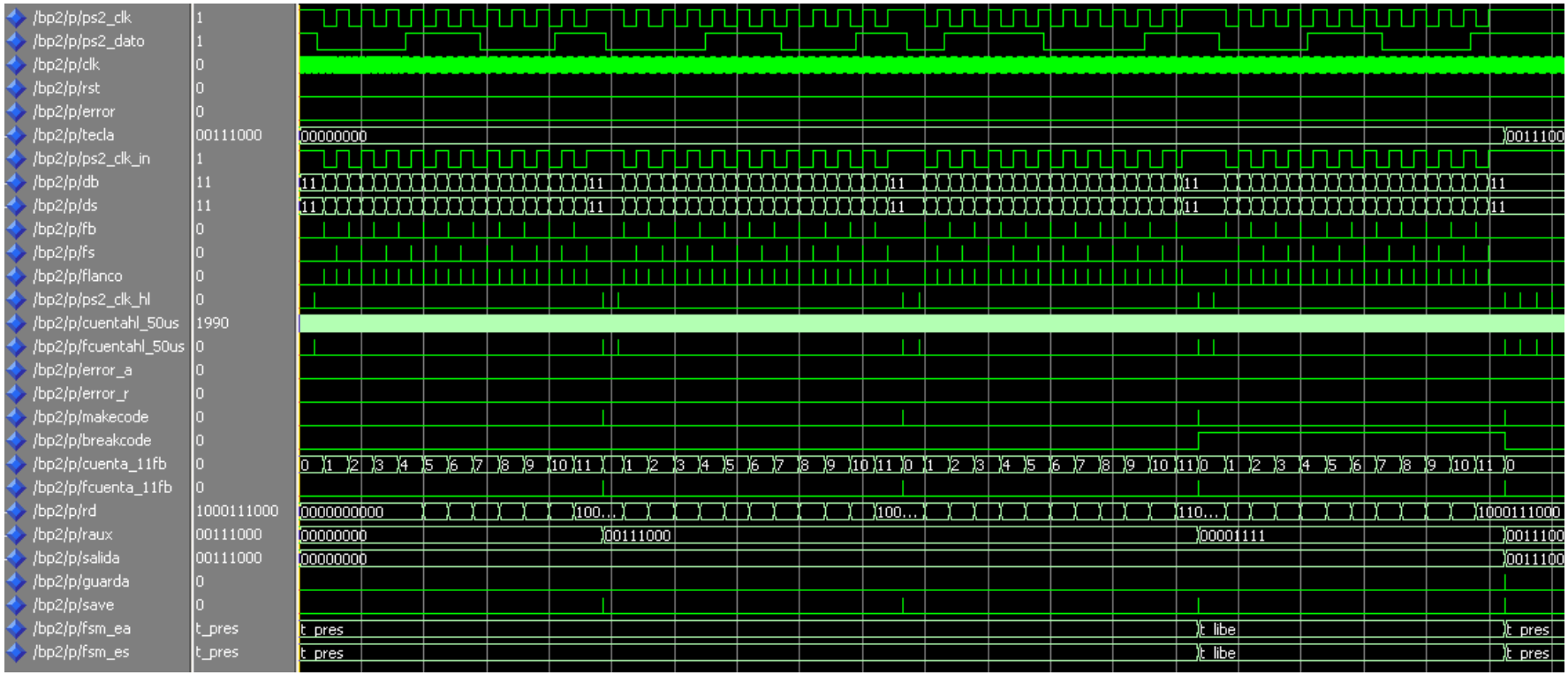


Figura 15. Simulación 2.

Esta simulación muestra el funcionamiento correcto de la transmisión de la tecla *'A'* al igual que la simulación anterior, pero en este caso la secuencia de sucesos es la siguiente: 1- se envía el *makecode* de la tecla *'A'* <1C> 2- se vuelve a enviar el *makecode* <1C> 3- se envía el *breakcode* <F0><1C>. Se observa que en este caso la señal *makecode* se activa 4 veces y que en la salida *tecla* no se registra el *makecode* hasta que se activa la señal de *guarda*.

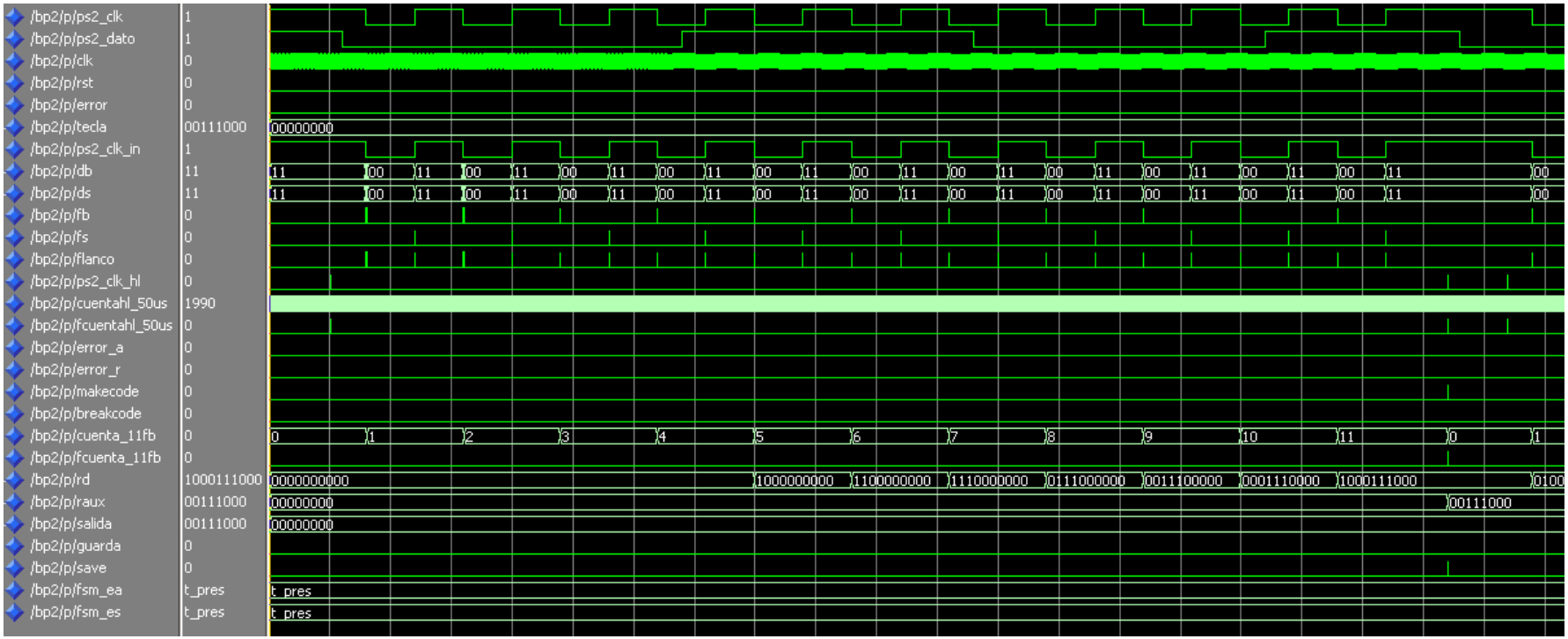


Figura 16. Simulación 3.

Esta simulación está hecha específicamente para explicar el funcionamiento del registro de desplazamiento de 10 bits y que además no es necesario que sea de 11bits. Observando la figura y centrándose en el envío 10, en ese instante el RD tiene un valor de "0001110000 b". En el siguiente envío este valor cambia a "1000111000 b" y es en ese momento donde se activa la señal *save* y se guarda el valor de 8 bits correspondiente a la trama formada por la concatenación del segundo al noveno bit del registro de desplazamiento en el *raux* (registro auxiliar).

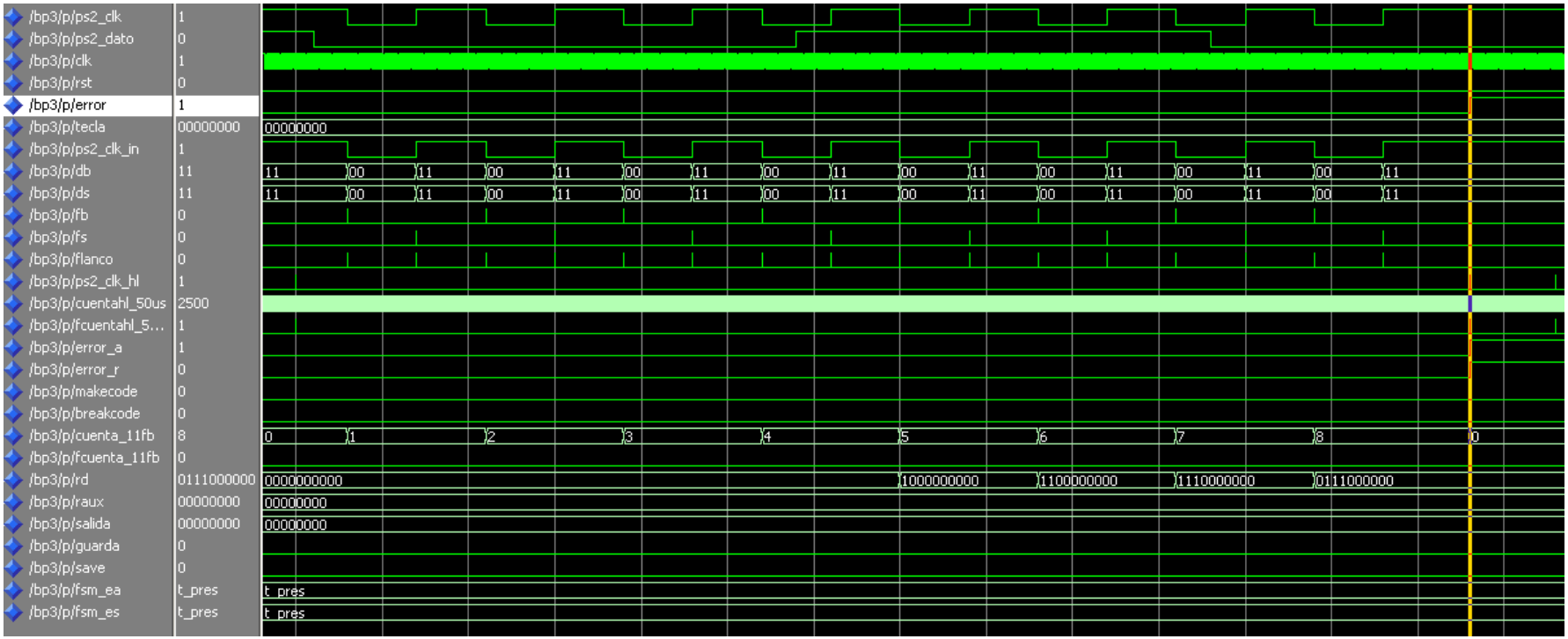


Figura 17. Simulación 4.

Esta simulación es para comprobar el funcionamiento de la detección de errores; específicamente en este caso se observa que la señal *error* se activa cuando se interrumpe la transmisión en el envío 8 y no se completan los 11 bits de una trama. Al detectar que la señal de *error* se pone a nivel alto habría que hacer un *reset* manual para que todo se reiniciara desde el primer estado y todos los registros se borraran, ya que si no se mantendría a *'1'* la señal *error*.

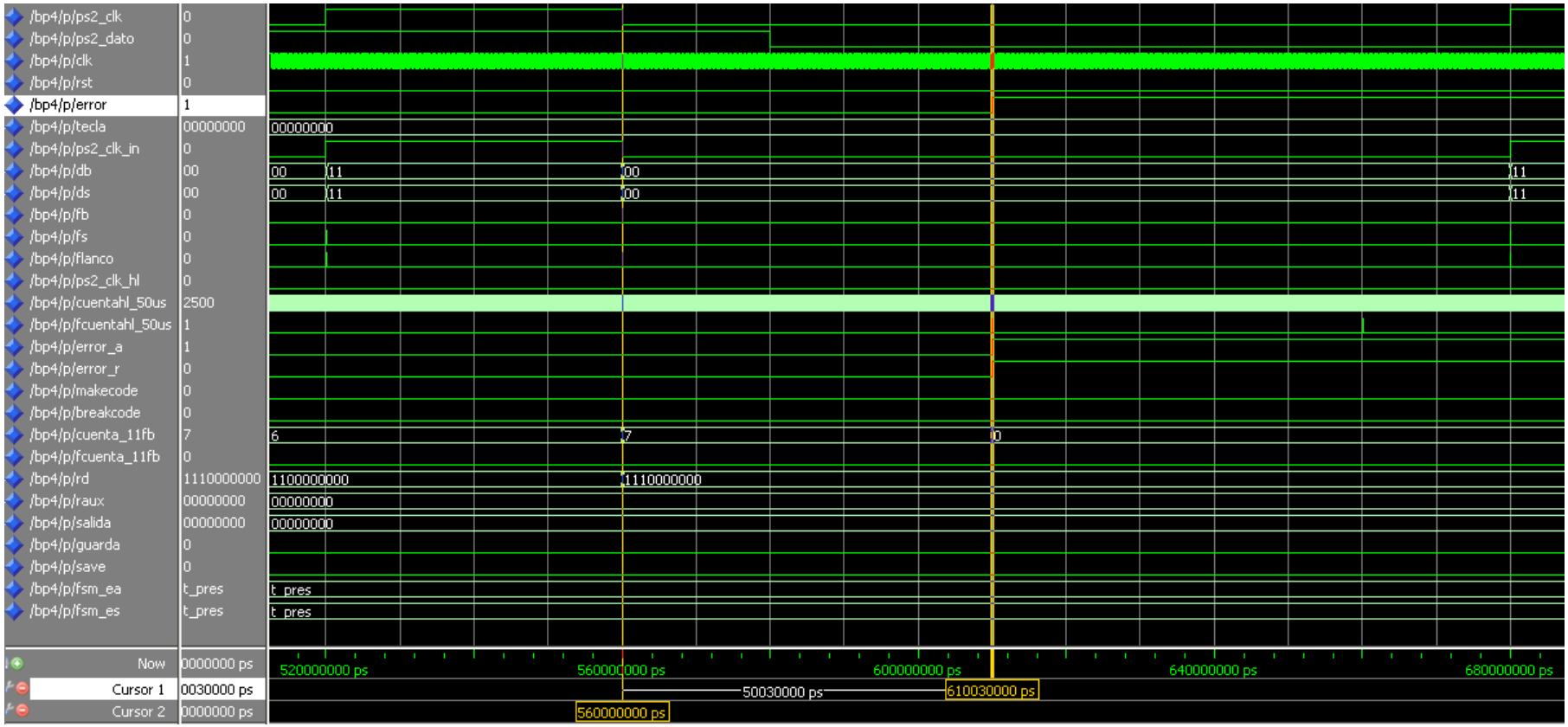


Figura 18. Simulación 5.

Esta simulación persigue el mismo objetivo que la anterior, pero en este caso se comprueba que la señal *error* se activa cuando la línea de reloj del teclado se mantiene a nivel bajo durante más de 50 µs. Esto significa, que esta línea está teniendo un comportamiento raro o no es el esperado, lo que demuestra que no se está cumpliendo con el protocolo de comunicaciones del puerto *PS/2*.

10. Bloque 2: Interfaz y protocolo de comunicaciones de la pantalla VGA.

10.1. Descripción del problema.

En este bloque 2 se desarrollará el protocolo a interfaz de comunicaciones *VGA*, por este motivo es necesario dominar conceptos e información sobre, ciertos aspectos. El primero de ellos es cómo funciona el puerto *VGA* y su configuración física. El siguiente aspecto que es importante saber es acerca del funcionamiento de una pantalla *VGA* y las especificaciones técnicas que influyan en nuestro desarrollo. Otro dato que hay que estudiar al detalle es todo lo que tenga que ver con la temporización de las señales de este puerto ya mencionado. Por último y como en el bloque anterior conocer cuáles son las características de la *FPGA* que usaremos en este bloque.

10.2. Puerto VGA.

La *Spartan 3E-Starter Board* cuenta con un puerto de pantalla *VGA* mediante el conector *DB-15*, cuya interfaz física es la que se muestra en la Figura 19. Es un conector que está conformado por 15 pines divididos en tres grupos o filas de 5 pines cada una [ver Tabla 11.]. Las patillas que nos interesan para el diseño del protocolo de comunicaciones *VGA* son cinco, exactamente estamos hablando de la *1*, *2*, *3*, *13* y *14*. De la *1* a la *3* son las señales de color, en este caso sería señal *VGA Roja*, señal *VGA Verde* y señal *VGA Azul* respectivamente. Luego están los pines *13* y *14* que básicamente son las señales de sincronismo de pantalla, estas serían la sincronización horizontal *HSync* y la sincronización vertical *VSync* respectivamente (26).

DB-15 (VGA)	Señal
1	Canal Rojo (RED)
2	Canal Verde (GREEN)
3	Canal Azul (BLUE)
4	No Implementado
5	No Implementado
6	Tierra (GND)
7	Tierra (GND)
8	Tierra (GND)
9	Alimentación (V_{CC} = +5V)
10	Tierra (GND)
11	Tierra (GND)
12	No Implementado
13	Sincronización Horizontal (HSync)
14	Sincronización Vertical (VSync)
15	No Implementado

Tabla 11. Pines del puerto *VGA* (26).

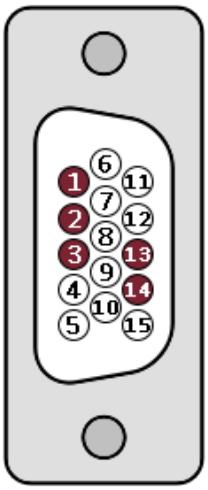


Figura 19. Conector DB-15 *VGA* (26).

La placa *Spartan-3E* acciona directamente las cinco señales *VGA* que se utilizarán mediante de resistencias [ver Figura 20.]. Cada canal de color tiene una resistencia en serie de valor 270 Ω ; esta resistencia en serie, en combinación con la impedancia aproximada de 75 Ω que existe en el cable *VGA*, asegura que las señales de color permanecen en el rango especificado para *VGA* de 0 a 0.7 V. Las señales *HSync* y *VSync* utilizan también resistencias en serie, pero en este caso de valor 82.5 Ω. Las líneas de color representan un bit cada una, para

Canal Rojo (RED), *Canal Verde (GREEN)* y *Canal Azul (BLUE)* respectivamente, su combinación de estados alto o bajo (*'0'* o *'1'*) es lo que genera los ocho colores posibles (2^3=8) mostrados en la Tabla 12 (26).

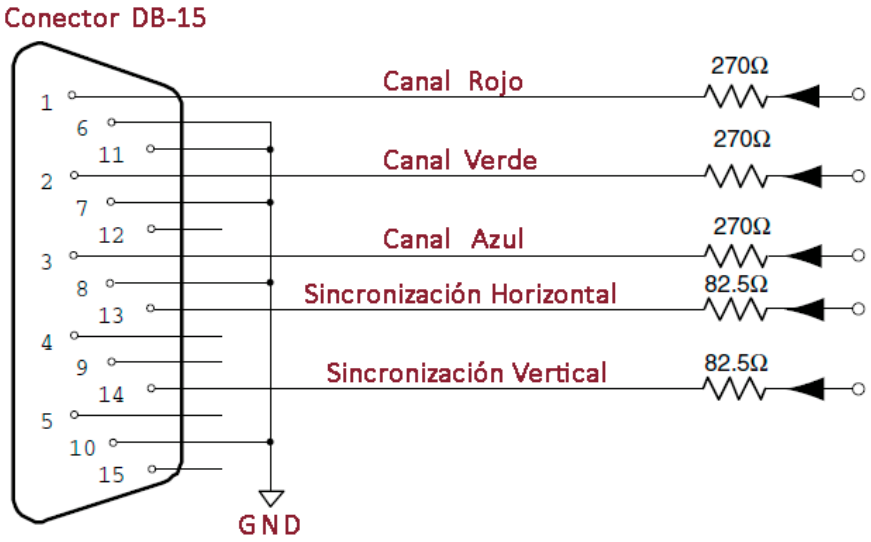


Figura 20. Conexiones del puerto *VGA* de la Placa *Spartan 3E-Starter Board* (26).

Canal Rojo	Canal Verde	Canal Azul	Color Resultante
0	0	0	Negro
0	0	1	Azul
0	1	0	Verde
0	1	1	Turquesa
1	0	0	Rojo
1	0	1	Magenta
1	1	0	Amarillo
1	1	1	Blanco

Tabla 12. Código de colores formados por 3 bits (26).

10.3. Pantalla/Monitor VGA.

Las pantallas *VGAs* basadas en la tecnología *CRT* utilizan haces de electrones en movimiento, de amplitud modulada (o rayos catódicos) para mostrar la información en una pantalla recubierta de fósforo. Las que son basadas en tecnología *LCD*, utilizan una serie de interruptores que pueden imponer una tensión a través de una pequeña cantidad de cristal líquido, cambiando de este modo la permisividad de luz a través del cristal sobre una base de píxel por píxel. Aunque la siguiente descripción va dirigida a las pantallas con tecnología *CRT*, las pantallas *LCD* han evolucionado para usar las mismas temporizaciones en las señales que las pantallas *CRT*. En consecuencia, todo lo expuesto a continuación es válido tanto a las pantallas *CRTs* como para las *LCDs* (26).

10.3.1. Funcionamiento de las señales VGA.

Dentro de una pantalla *CRT*, las ondas de corriente pasan a través de la bobina para producir campos magnéticos que desvían los haces de electrones transversalmente a la superficie de la pantalla siguiendo un patrón como trama. El movimiento es horizontalmente de izquierda a derecha y verticalmente de arriba a abajo. Como se muestra en la Figura 21, la información sólo se muestra cuando el haz se mueve en la dirección de avance de izquierda a derecha y de arriba a abajo, y no durante el tiempo que el haz vuelve de nuevo a la orilla izquierda o la parte superior de la pantalla. Por lo tanto, gran parte del tiempo de visualización potencial se pierde en períodos de supresión o retorno, que es cuando el haz se pone a cero y se estabiliza para comenzar un nuevo pase de visualización horizontal o vertical (26).

En el caso de una pantalla *LCD*, el funcionamiento es similar y utilizan las mismas temporizaciones en las señales con la diferencia que esta tecnología no usa haces de electrones; sino que usan la variación de tensión mediante interruptores sobre una pequeña cantidad de cristal líquido. De esta forma se controla el paso de luz a través del cristal de cada píxel. Para un monitor *LCD e*l muestreo en pantalla o dibujo de los pixeles también es horizontal de izquierda a derecha y verticalmente de arriba a abajo. Como se muestra en la Figura 21. Aquí también la información sólo se muestra cuando el haz se mueve en la dirección de avance de izquierda a derecha y de arriba a abajo, y no durante el tiempo que el haz vuelve de nuevo a la orilla izquierda o la parte superior de la pantalla. Por lo tanto, gran parte del tiempo de visualización potencial se pierde en períodos de supresión o retorno, que es cuando el haz se pone a cero y se estabiliza para comenzar un nuevo pase de visualización horizontal o vertical (26).

La resolución de la pantalla (en nuestro caso 640x480 pixeles) define el tamaño de los haces de electrones, que dependen de la frecuencia a la que se trazan los haces a través de la pantalla y de la frecuencia a la que el haz de electrones es modulado; en el caso de los monitores *CRT*. En el caso de monitores *LCD*, la resolución de pantalla define el tamaño de estas pequeñas cantidades de cristal líquido y que dependen de la frecuencia con la que se varía la tensión de los interruptores para controlar la permisividad de la luz. Las pantallas *VGA* modernas soportan múltiples resoluciones de pantalla, y el controlador *VGA* es el que dicta la resolución a usar, mediante la producción de señales de temporización para controlar los patrones de la trama. El controlador produce pulsos de sincronización que establecen la frecuencia en la que la corriente fluye a través de las bobinas de deflexión y asegura que los datos de cada píxel o de vídeo, se aplican a los cañones de electrones en el momento correcto (26).

Los datos de vídeo normalmente provienen de una memoria de actualización de vídeo con uno o más bytes asignados a cada posición de píxel. La *FPGA* junta y utiliza tres bits por píxel, produciendo uno de los ocho posibles colores que se muestran en la Tabla 12. El controlador indexa en el búfer de datos de vídeo como los haces se mueven por la pantalla o como se varían las tensiones de los interruptores del *LCD* y el controlador entonces recupera y aplica los datos de vídeo a la pantalla precisamente en el momento en el que el haz de electrones se mueve a través de un píxel dado o se cambia la permisividad de la luz para un pixel dado (26).

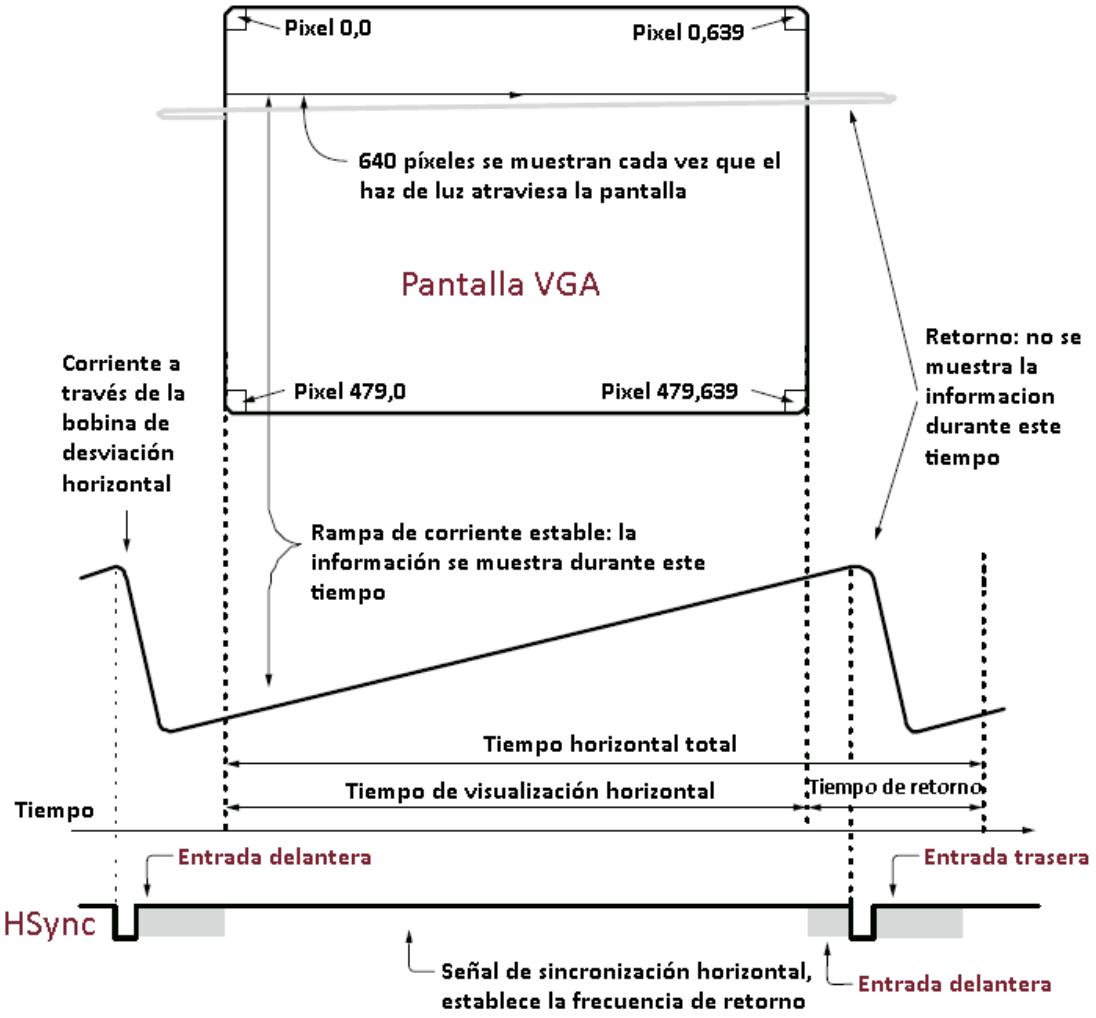


Figura 21. Funcionamiento de las señales en una pantalla *VGA* (26).

En la Figura 22, se muestra como el controlador *VGA* genera las señales de sincronización horizontal *HSync* y vertical *VSync* y coordina la entrega de datos de vídeo por cada reloj de píxel. El reloj de píxel define el tiempo disponible para mostrar un píxel de información. La señal *VSync* define la frecuencia de actualización de la pantalla, o la frecuencia con la que toda la información que hay en la pantalla se vuelve a dibujar. La frecuencia mínima de actualización varía en función de la pantalla de fósforo y la intensidad del haz de electrones, con frecuencias de actualización prácticas en el rango de 60 Hz a 120 Hz. El número de líneas horizontales que aparecen a una frecuencia de actualización dada define la frecuencia de retorno horizontal (26).

10.3.2. Protocolo VGA.

Los tiempos de las señales de la Tabla 13 son los especificados por el protocolo *VGA* para una pantalla cuya resolución es de 640x480 pixeles con un reloj por pixel de 25 MHz y 60 Hz ± 1 de frecuencia de actualización de la pantalla. La Figura 22 muestra la relación entre cada uno de los símbolos de tiempo. El tiempo para el ancho de pulso de sincronización T_{PW} y los intervalos de entradas delantera y trasera T_{FP} y T_{BP} se basan en observaciones de varias pantallas *VGA*. Los intervalos de entrada delantera y trasera son los tiempos de pulso pre y post sincronización; la información no se puede visualizar durante estos tiempos. El tiempo de visualización T_{DISP} es en el que se muestra la información en pantalla y el tiempo de pulso de

sincronización T_S es el tiempo total que demora en completarse un barrido horizontal o barrido de pantalla (26).

Símbolo	Parámetro de tiempo	*VSync*			*HSync*	
		Tiempo	Ciclos	Líneas	Tiempo	Ciclos
T_S	Pulso de sincronización	16.7 ms	416800	521	32 µs	800
T_{DISP}	Visualización	15.36 µs	384000	480	25.6 µs	640
T_{PW}	Ancho de pulso	64 µs	1600	2	3.84 µs	96
T_{FP}	Entrada delantera	320 µs	8000	10	640 ns	16
T_{BP}	Entrada trasera	928 µs	23200	29	1.92 µs	48

Tabla 13. Tiempos del protocolo *VGA* para una resolución 640x480 (26).

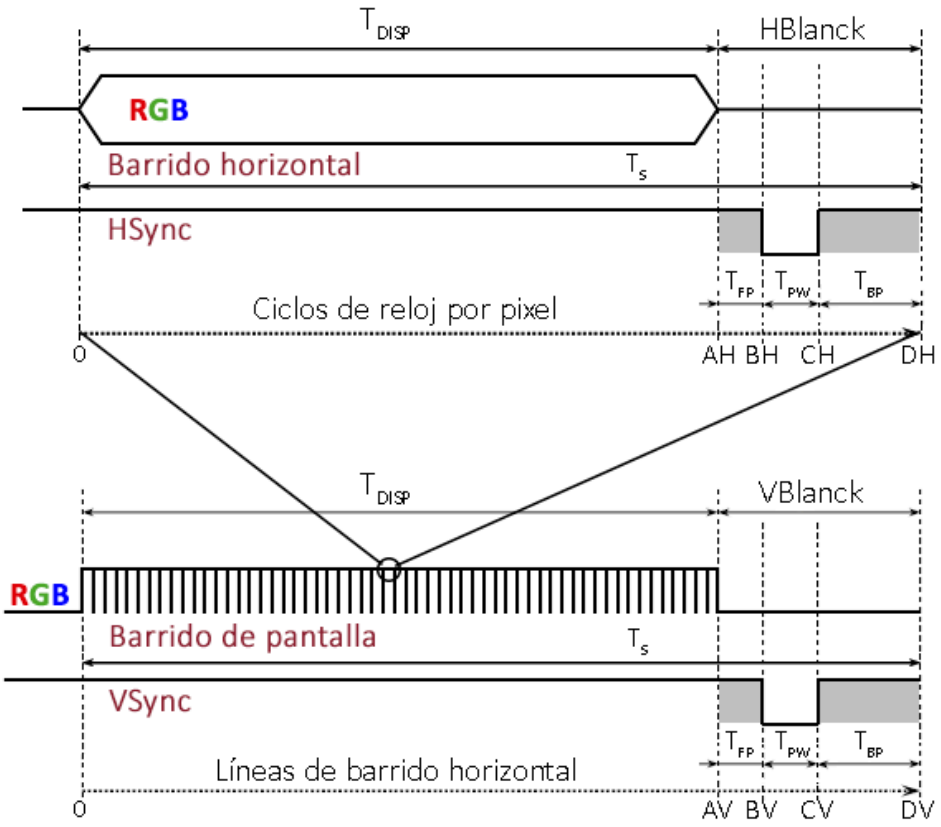


Figura 22. Formas de onda y temporización de los barridos horizontal, de pantalla, *HSync* y *VSync* (27).

En general, una pantalla *VGA* se controla por medio de cinco señales analógicas, tres que especifican la intensidad de los colores (*RGB*, *Red-Green-Blue*) y dos señales de sincronismo, horizontal y vertical (*HSync* y *VSync*). Para dibujar una imagen en el monitor, hay que realizar un barrido de líneas, empezando por la esquina superior izquierda y siguiendo hacia la derecha y hacia abajo. Es decir, el haz de electrones va dibujando los datos en líneas horizontales. Después de cada línea horizontal es necesario dar un pulso a la señal de sincronismo horizontal *HSync* y después de cada pantalla, cuando están dibujadas todas las líneas horizontales, hay que dar un pulso a la señal de sincronismo vertical *VSync*. En la Figura 22 se aprecian las formas de onda de un barrido de línea horizontal (27).

Para implementar el barrido de cada línea horizontal se utiliza un contador, que se reiniciará al comienzo de cada línea y que permitirá temporizar la señal *HSync* apropiadamente. Este contador rastrea la ubicación actual de la visualización de píxeles en una

fila determinada. El proceso de barrido de línea se repite tantas veces como líneas haya en la pantalla y después se procede a realizar el sincronismo vertical (27).

Un contador aparte controla la sincronización vertical, que se reiniciará al comienzo de cada pantalla y que permitirá temporizar la señal *VSync*. Este contador incrementa con cada pulso de *HSync*. La señal *VSync* tiene una forma equivalente a *HSync*, pero el pulso se produce tras haberse barrido todas las líneas horizontales, como se observa en la Figura 22. Este contador rastrea la fila actual de la pantalla (27).

Estos dos contadores de funcionamiento continuo crean la dirección de visualización de vídeo, es decir, ubican constantemente el píxel que se visualizará. No se especifica una relación de tiempo entre el inicio del pulso de *HSync* y el inicio del pulso de *VSync*. En consecuencia, los contadores se pueden organizar para formar fácilmente las direcciones de vídeo, o para minimizar la lógica de decodificación para la generación de impulsos de sincronización (27).

Debido a la anterior característica, como la frecuencia de actualización es de 60 Hz y para una frecuencia de reloj de 25 MHz (la mitad del reloj de la placa de *Xilinx*), los tiempos de sincronismo correspondientes a cada evento y los valores correspondientes de los contadores se muestran en la Tabla 14 y que se pueden ver en la Figura 22 (27).

HSync			***VSync***		
Símbolo	**Tiempo**	**Pixeles**	**Símbolo**	**Tiempo**	**Líneas horizontales**
AH	25.60 µs	640	**AV**	15.360 ms	480
BH	26.24 µs	656	**BV**	15.680 ms	490
CH	30.08 µs	752	**CV**	15.744 ms	492
DH	32.00 µs	800	**DV**	16.672 ms	521

Tabla 14. Tiempos de sincronismo del diseño *VGA* (27).

10.4. Especificaciones del bloque 2.

10.4.1. Interfaz.

En este apartado se especifican cuáles son las entradas y salidas (E/S) de este bloque 2. Las mismas se pueden ver en el anexo de esta memoria 16.2 más adelante. Las señales de este bloque son:

Entradas

- **clk**: señal que representa el reloj de la *FPGA*, el cual fija todo el diseño. Tiene una frecuencia de 50 MHz lo que produce un periodo de 20 ns.
- **rst**: señal de inicialización asíncrona del sistema, activa a nivel alto.
- **Color:** señal que indica el código de 3 bits que designa el color a mostrar en cada píxel que se vaya a procesar.

Salidas

- **HSync**: señal de sincronismo horizontal, la cual se activa a los 656 ciclos de reloj siguiendo el protocolo *VGA*, activa a nivel bajo.
- **VSync**: señal de sincronismo vertical, la cual se activa a los 416 771 ciclos de reloj siguiendo el protocolo *VGA*, activa a nivel bajo.
- **R, G, B:** señales de 1 bit, cuya función es seleccionar el color a mostrar en pantalla.
- **X:** señal de 10 bits que se utiliza para indicar la posición horizontal del barrido de la pantalla, es de 10 bits para poder alcanzar los 800 pixeles de ancho (2^10=1024).
- **Y:** señal de 10 bits para indicar la posición del barrido vertical de la pantalla, es de 10 bits para poder alcanzar los 521 pixeles de alto (2^10=1024).

10.4.2. Funcionalidad.

La idea de funcionamiento de la interfaz de una pantalla *VGA* es materializar o hacer gráfico los datos que vengan del bloque de teclado. A modo general es la manera de visualizar toda esa información que se introduce en el teclado. Es la interfaz gráfica de un sistema con la cual actúa cualquier usuario. Como se mencionó anteriormente en el diseño del teclado este desarrollo también es independiente de la tecnología, aunque se utilizara la placa de la Figura 23.

En este bloque de interfaz *VGA* se utilizarán las características siguientes de la placa de *Xilinx* [ver Figura 23.]. Igual que en el bloque anterior de la interfaz *PS/2*, la más importante y la que más se tendrá en cuenta para el desarrollo del protocolo actual es su *reloj principal*; cuyo valor de frecuencia es de 50 MHz, lo que implica que tenga un período de 20 ns. El pin asignado en la placa para este reloj se muestra en la Tabla 15. Para poder dibujar en una pantalla utilizaremos el puerto *VGA* que presenta la *Spartan*; de los 15 pines que dispone este conector solo utilizaremos cinco, *HSync*, *VSync*, *Rojo*, *Verde* y *Azul*, cuyas direcciones también se muestran en la Tabla 15. Como última característica a usar de la placa en este diseño, tenemos la implementación de un *Reset* para volver al estado inicial por si hubiera algún problema o posible desajuste de funcionamiento. Este botón fue asignado en este proyecto a un pin específico de la *FPGA*, cuya dirección viene en la Tabla 15.

Elemento	Pin
Reloj	C9
Reset	L13
HSync	F15
VSync	F14
Rojo	H14
Verde	H15
Azul	G15

Tabla 15. Pines de la *FPGA* para el bloque 2 (26).

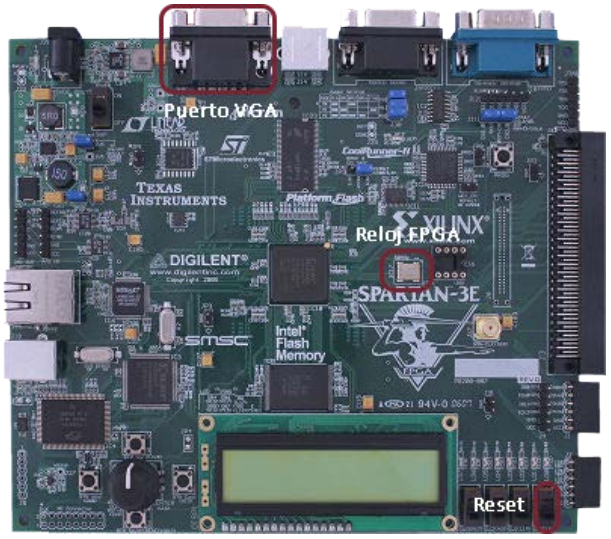


Figura 23. Características usadas de la placa *Spartan 3E-Starter Board* para el bloque 2 (26).

10.4.3. Arquitectura.

En esta sección solo se detallan las partes o bloques del circuito, que se explicará con detalle en el siguiente apartado 10.4.4. En la Tabla 16 se enumeran todos los bloques del circuito y en la Figura 24 se puede ver el diagrama completo de la arquitectura.

Bloque	Descripción
1	Biestable tipo 'T' [Divisor de frecuencia]
2	Contador de 10 bits para las 'X'.
3	Contador de 10 bits para las 'Y'.
4	Comparador en 'X'.
5	Comparador en 'Y'.
6	Blanqueador.

Tabla 16. Resumen de los bloques del bloque 2.

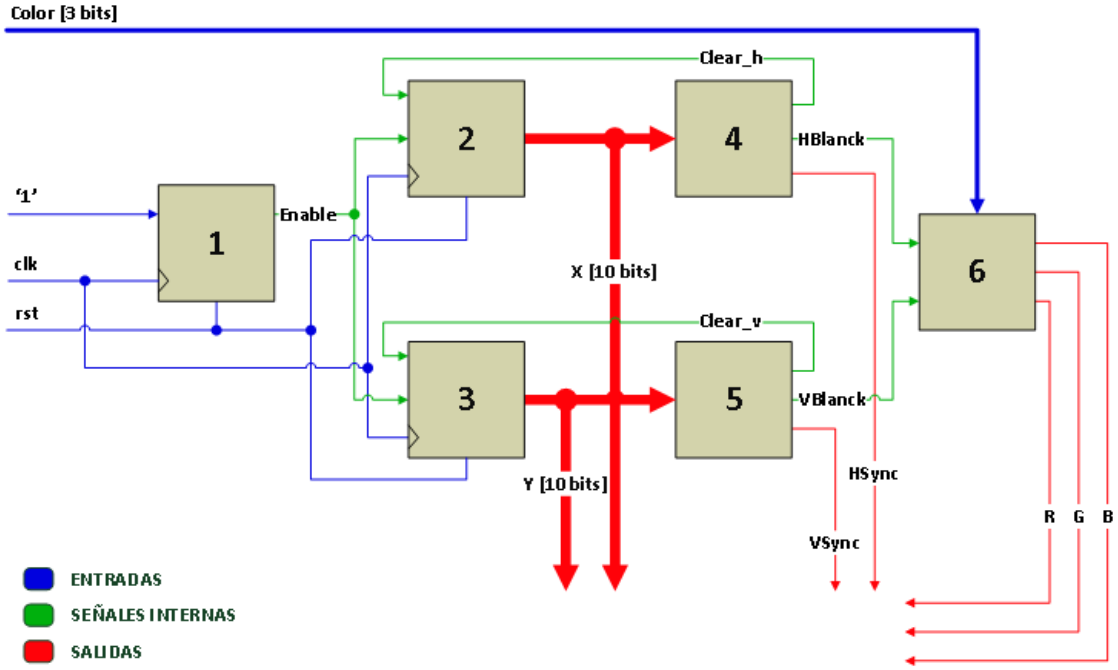


Figura 24. Diagrama de bloques del circuito del bloque 2.[4]

[4] En este diagrama de bloques se han numerado las cajas para facilitar la explicación y detalle a posteriori, aunque se sabe que deberían tener sus propios nombres.

10.4.4. Diseño detallado y síntesis del circuito.

Como se observa en la Tabla 16, el diseño del protocolo *VGA* se realizó mediante seis bloques de funcionamiento bien definidos y que en este apartado se explicará al detalle el comportamiento de cada uno. Conociendo que el protocolo se cumple para una frecuencia de actualización de 60 Hz con una frecuencia de reloj por pixel de 25 MHz, hubo que diseñar el bloque 1 cuya función es transformar los 50 MHz del reloj de la placa de Xilinx a los 25 MHz requeridos.

Este primer bloque es un Biestable tipo *'T'* y su funcionamiento básico es que cuando la entrada *'T'* toma el valor *'1'*, la salida *Q* cambia de estado (de *Q* a *Q+*). En el caso de que se mantenga la entrada *'T'* permanente a nivel alto o *'1'*, este biestable se comporta como un divisor de frecuencia de la señal de reloj entre dos. Su símbolo es como el de la Figura 25 y su ecuación característica que describe su comportamiento es $Q_+=T \oplus Q$. En la Tabla 17 se muestran todos sus posibles estados, pero en este caso solo usaremos los dos últimos, ya que usamos la configuración como un divisor de frecuencia.

T	Q	Q+
0	0	0
0	1	1
1	0	1
1	1	0

Tabla 17. Tabla de verdad de un Biestable tipo *'T'*.

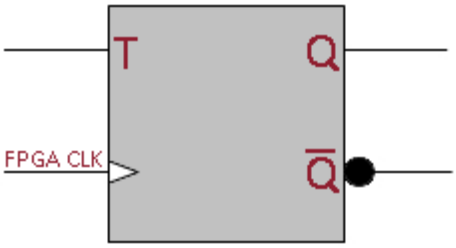


Figura 25. Biestable tipo *'T'*.

En la Figura 26 se detallan con exactitud las formas de onda del anteriormente explicado biestable *'T'*. En esta imagen se aprecia con claridad como por cada dos ciclos completos del reloj de la *FPGA* la salida *Q* solo completa un ciclo. Se ven los períodos de ambas señales y los tiempos de cada una. La señal *Q* que en nuestro diseño es la señal *Enable* [ver Figura 24.] es la que habilita todos los demás conjuntos para hacerlos funcionar a la frecuencia prefijada por el protocolo.

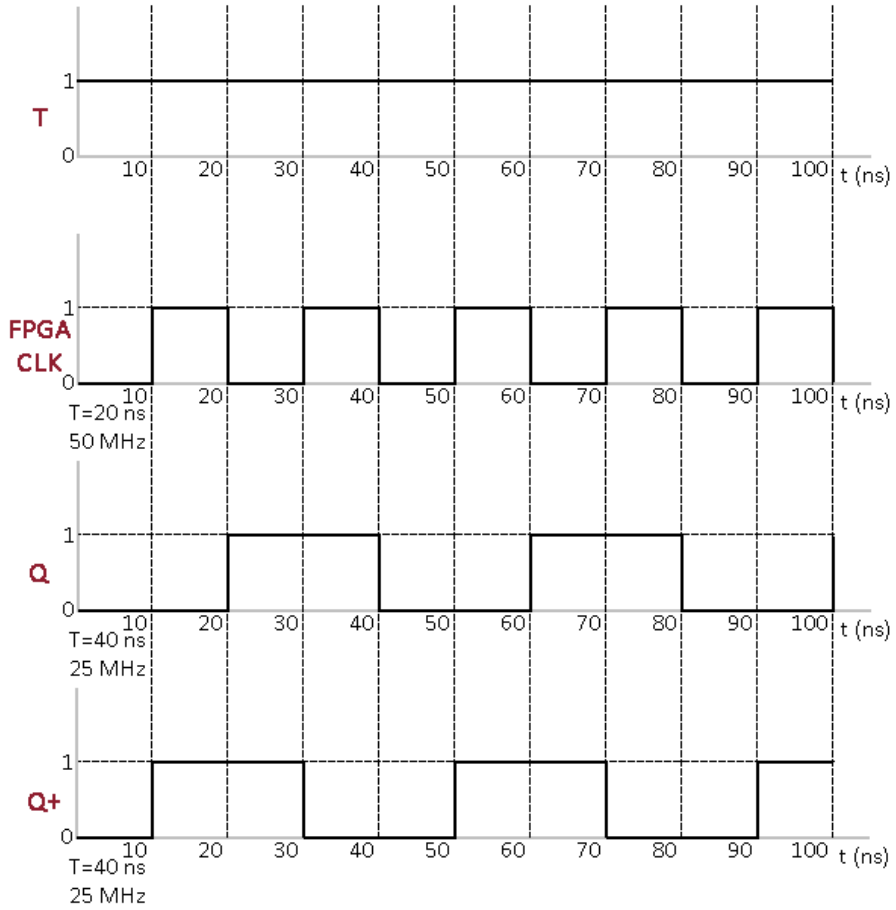


Figura 26. Formas de onda de un Biestable tipo *'T'*.

Para implementar los contadores, bloques 2 y 3, es necesario remitirse a la Tabla 14. En ella se puede ver que los puntos *AH* y *AV* son 640 y 480 respectivamente; lo que se corresponde con la resolución de pantalla o número de píxeles de alto por ancho de la pantalla. Además, se observa que los puntos *DH* y *DV* son 800 y 521 respectivamente, coincidiendo con la cantidad de pixeles que se recorren en total en una línea horizontal y la cantidad de líneas en recorrer verticalmente. Por tanto, para poder hacer los barridos ya sea horizontal como vertical, se implementaron estos contadores; cuyas salidas (valor de la cuenta de cada uno) representarán las coordenadas *X* e *Y* del píxel que se está representando en un momento dado.

Nótese que ambos contadores son de 10 bits; esto es debido a que en el caso del contador en *X* la cuenta debe alcanzar el valor 800, lo cual solo se consigue con $2^{10} = 1024$. Lo mismo sucede para el contador en *Y,* el cual debe alcanzar el valor de cuenta 521. No se podría usar contadores de menos bits (por ejemplo de 9 bits), porque $2^9 = 512$ y no alcanzaríamos a barrer tan siquiera la totalidad de las líneas horizontales. Como se puede observar en la Figura 24, los contadores se reinician unas señales de *Clear_h* y *Clear_v* que provienen de los bloques 4 y 5 respectivamente.

Estos bloques se comportan como comparadores, es decir, su función consiste en generar tres señales claves para el cumplimiento del protocolo *VGA*. Estas señales son la de sincronismo (*HSync*, *VSync*, dependiendo de si es en *X* o en *Y*), la señal de blanqueo (*HBlanck*, *VBlanck*, dependiendo de si es en *X* o en *Y*) de la cual se hablará más adelante; y además la señal de reinicio de los contadores (*Clear_h*, *Clear_v*, dependiendo de si es en *X* o en *Y*). La señal de sincronismo es la que controla que se haya completado cada barrido (horizontal y

vertical), la señal de blanqueo son las que indican al bloque blanqueador cuando se éste funciona y las de reinicio de los contadores son las que borran la cuenta con cada señal de sincronismo.

El bloque 6 y último, es el encargado como su propio nombre indica de borrar los colores de la pantalla. Para que se entienda mejor, lo que hace este bloque es fijar exactamente la resolución de la pantalla. Una vez que se alcanzan los 640 pixeles en el sentido horizontal el resto de pixeles hasta llegar a los 800 se dibujan en negro; lo mismo sucede con las 480 líneas en sentido vertical hasta llegar a las 521 que se pintan en negro. Cuando se activa la señal de blanqueo (*HBlanck* y *VBlanck*) a nivel alto o *'1'* es cuando las salidas del bloque 6 *R*, *G*, *B* se ponen a nivel bajo o *'0'*, generando el color negro como se aprecia en la Tabla 12.

El diseño realizado hay que sintetizarlo en el programa de *Xilinx ISE Project Navigator*. Este software de *Xilinx* se encarga de verificar, depurar, corregir la sintaxis e implementar el código hecho. Cuando se transfiere a la *FPGA* dicho código ya sintetizado e implementado. El *Project Navigator* nos aporta información importante sobre el diseño; dentro de esta información tenemos una serie de informes, los cuales nos brindan ciertos parámetros importantes e indicadores del diseño actual. Con estos datos podemos conocer la calidad de nuestro diseño o cuán rápido es su funcionamiento o simplemente cuanto ocupa dentro de la *FPGA*.

El resumen de utilización es uno de los informes más usados; de éste podemos extraer información acerca del número de unidades lógicas que ocupa el diseño del total disponible en la *FPGA*, ya sean *Flip-Flops*, *LUTs* o *Slices*. Toda esta información se ofrece en cantidades usadas, disponibles y en los porcentajes que representan. En la Figura 27 se muestra este resumen del diseño del controlador o driver de la pantalla *VGA* final y funcional.

Device Utilization Summary			
Logic Utilization	**Used**	**Available**	**Utilization**
Number of Slice Flip Flops	21	9,312	1%
Number of 4 input LUTs	41	9,312	1%
Number of occupied Slices	32	4,656	1%
Number of Slices containing only related logic	32	32	100%
Number of Slices containing unrelated logic	0	32	0%
Total Number of 4 input LUTs	59	9,312	1%
Number used as logic	41		
Number used as a route-thru	18		
Number of bonded IOBs	30	232	12%
Number of BUFGMUXs	1	24	4%
Average Fanout of Non-Clock Nets	2.86		

Figura 27. Resumen de la utilización de la *FPGA* bloque 2.

Como se muestra en la captura anterior, el resultado obtenido del análisis es que el número de *Slices* ocupados es 32 de 4656 disponibles, lo que equivale al 1% de utilización. El número de *Flip-Flops* es 21 de 9312 lo que equivale al 1% de utilización. El número de *4 input LUTs* es 59 de 9312 que corresponde al 1% del total disponible. El número de *bonded IOBs* es 30 de 232 que es el 12%. El número de *BUFGMUXs* es 1 de 24 que es el 4% del total.

Otro de los informes importantes del *Project Navigator* es el Resumen de Tiempos; este nos ofrece datos significativos a tener en cuenta cuando se quiera utilizar este diseño en otras placas o *FPGAs*. En este caso se obtuvieron los resultados mostrados en la Tabla 18:

Minimum period: 6.369ns (Maximum Frequency: 157.011MHz)
Minimum input arrival time before clock: No path found
Maximum output required time after clock: 8.113ns
Maximum combinational path delay: 6.209ns

Tabla 18. Extracto del informe de tiempos de la *FPGA* bloque 2.

En la Tabla 18, se muestra que el período mínimo de una señal es de 6,369 ns, a una frecuencia máxima de 157,011 MHz. Esto se traduce en que si se utilizara este diseño con otra *FPGA* habría que asegurarse que ésta tiene un reloj con una frecuencia igual o menor que la frecuencia calculada, debido a que si el período fuese menor no funcionaría el circuito. Otro dato que aporta este informe es el tiempo mínimo que demora en llegar una señal de entrada antes de un pulso de reloj; en este caso el resultado es *No path found* (implica que las señales de entrada no dependen de la señal de reloj) y el tiempo máximo que requiere una señal de salida después de un pulso de reloj es de 8,113 ns. Además informa que el retraso máximo en una ruta combinacional es de 6,209 ns. Toda esta información es crucial a la hora de usar este diseño en otras *FPGAs*. Es imprescindible cumplir con los tiempos y las frecuencias especificadas anteriormente para asegurar un funcionamiento correcto.

10.4.5. Descripción del banco de pruebas del bloque 2.

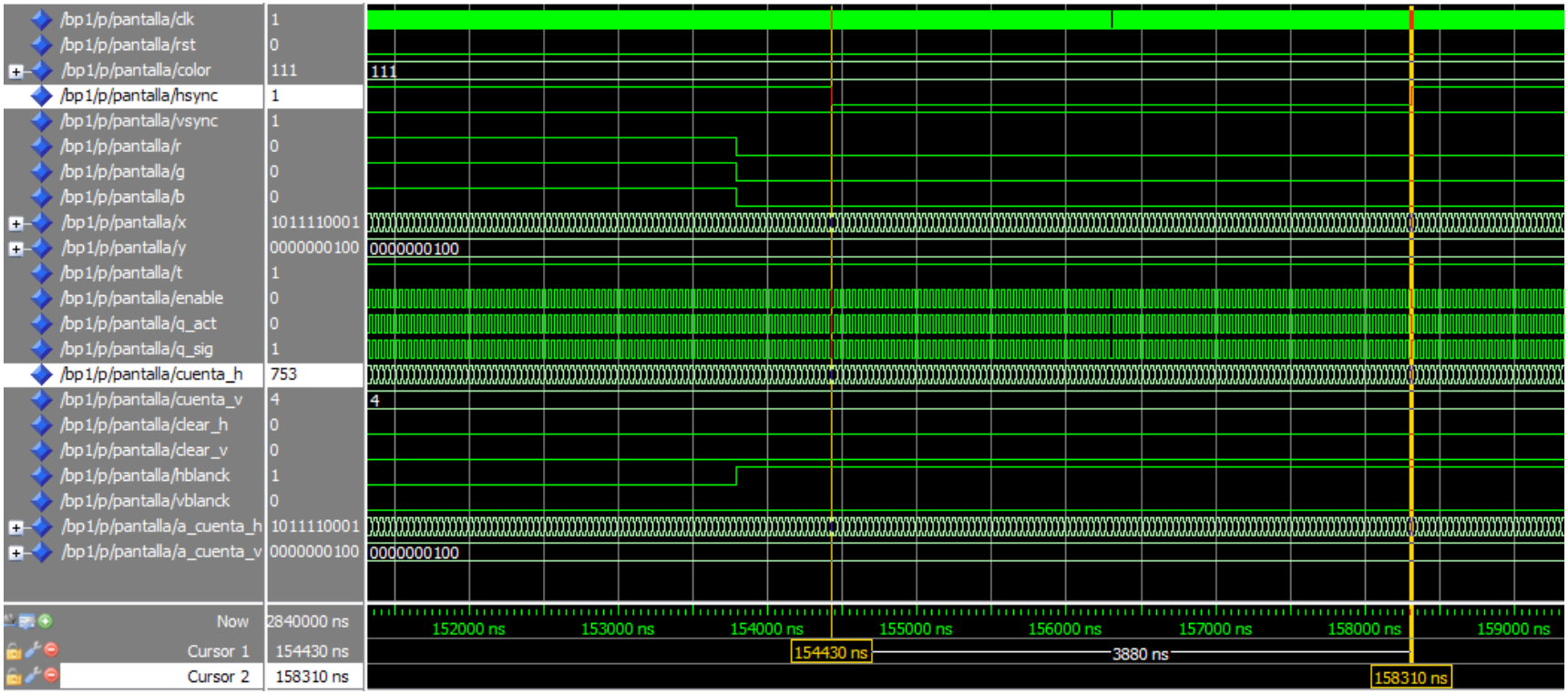


Figura 28. Simulación 6.

Con esta simulación se demuestra que el diseño cumple con la temporización de la señal de sincronización horizontal *HSync*. Esta señal es activa a nivel bajo *'0'* y se aprecia como durante unos 3880 ns permanece a nivel bajo. Según la Tabla 14 esta señal debe activarse en el píxel 656 (contabilizado a través de la señal *cuenta_h*) y mantenerse hasta terminar el píxel 752 (contabilizado a través de la señal *cuenta_h*), es decir, tiene que estar activada durante 97 píxeles (753-656=97) y si el diseño trabaja con un período de 40 ns se obtienen estos 3880 ns (97*40 ns=3880 ns).

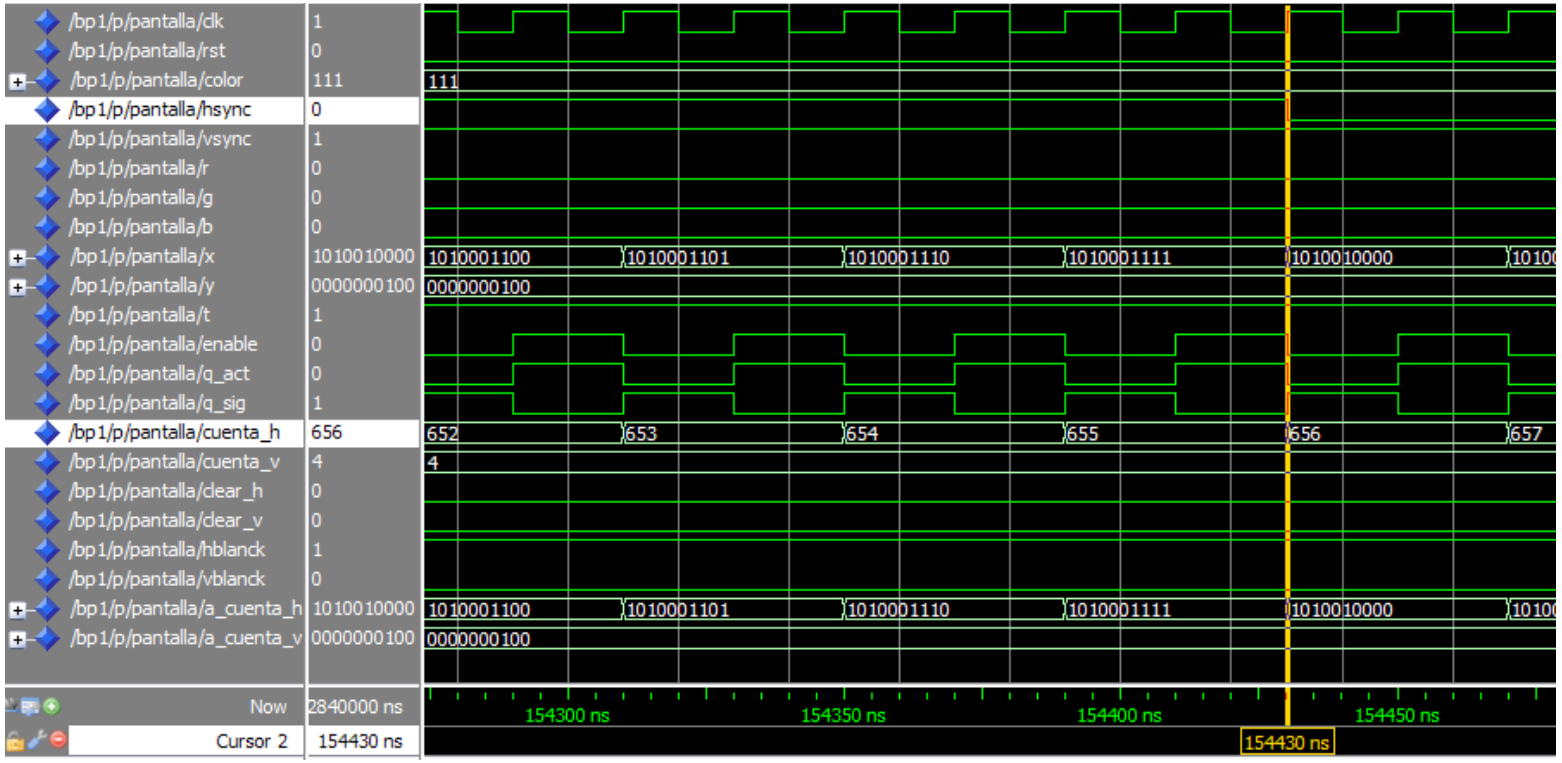


Figura 29. Simulación 7.

En esta simulación se muestra con más detalle cuando se activa la señal *HSync*. Se aprecia que cuando la señal *cuenta_h* (encargada de contar los píxeles cuyo período es de 40 ns) llega al píxel 656, *HSync* se pone a *'0'*. De esta forma se justifican los cálculos hechos en la simulación anterior [ver Figura 28].

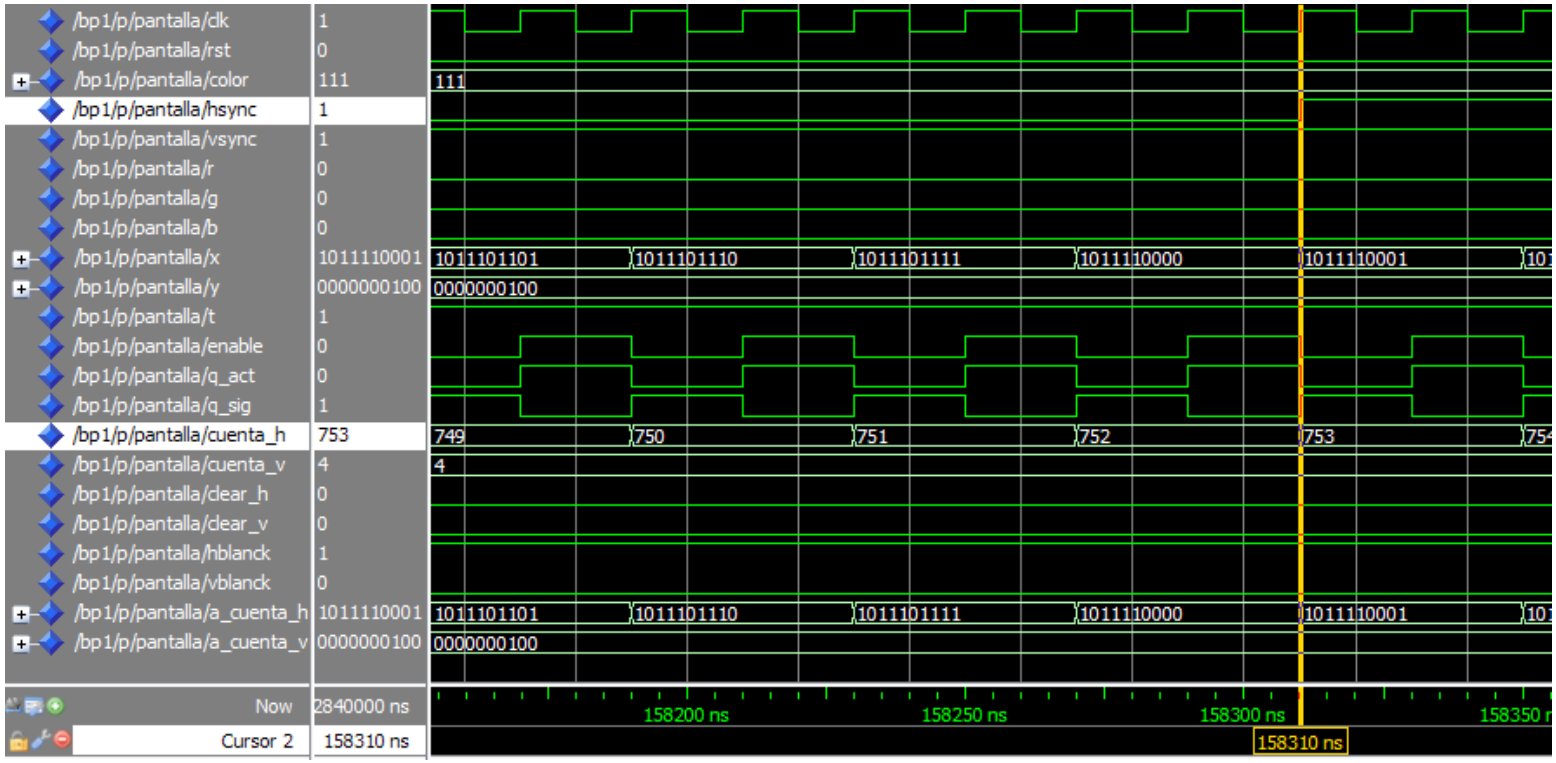


Figura 30. Simulación 8.

En esta simulación se muestra con más detalle cuando se desactiva la señal *HSync*. Se aprecia que cuando la señal *cuenta_h* (encargada de contar los píxeles cuyo período es de 40 ns) termina el píxel 752, *HSync* se pone a *'1'*. De esta forma se justifican los cálculos hechos en la simulación anterior [ver Figura 28].

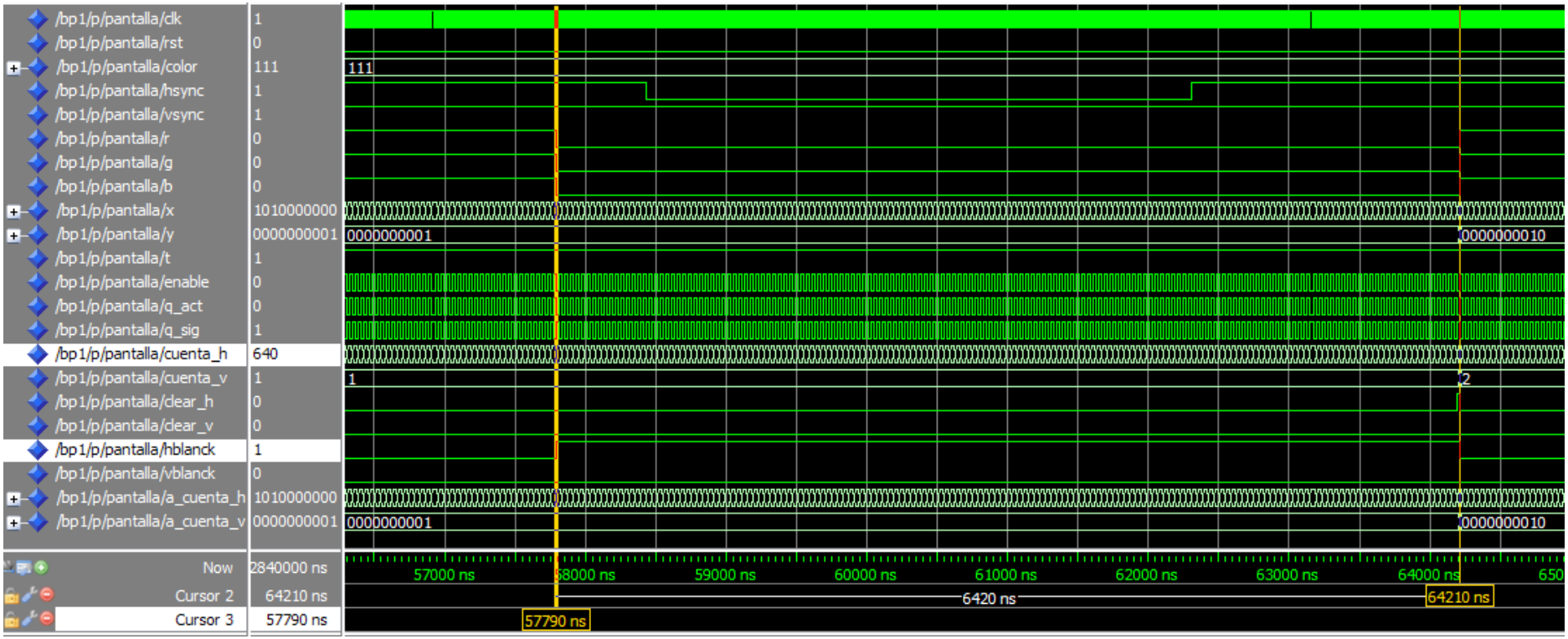


Figura 31. Simulación 9.

Con esta simulación se demuestra que el diseño cumple con la temporización de la señal de blanqueo horizontal *HBlanck*. Esta señal es activa a nivel alto *'1'* y se aprecia como durante unos 6420 ns permanece a nivel alto. Según la Tabla 14 esta señal debe activarse en el píxel 640 (contabilizado a través de la señal *cuenta_h*) y mantenerse hasta terminar el píxel 800 (contabilizado a través de la señal *cuenta_h*), es decir, tiene que estar activada durante 161 píxeles (801-640=161) y si el diseño trabaja con un período de 40 ns se obtienen unos 6440 ns (161*40 ns=6440 ns). Vemos que en la simulación faltan unos 20 ns (6440 ns-6420 ns=20 ns) y esto se debe a que el ciclo 800 de la señal *cuenta_h* se interrumpe por la mitad porque la señal *clear_h* se activa cuando se llega a ese punto y se utiliza para reiniciar el contador de *cuenta_h* [ver Figura 34].

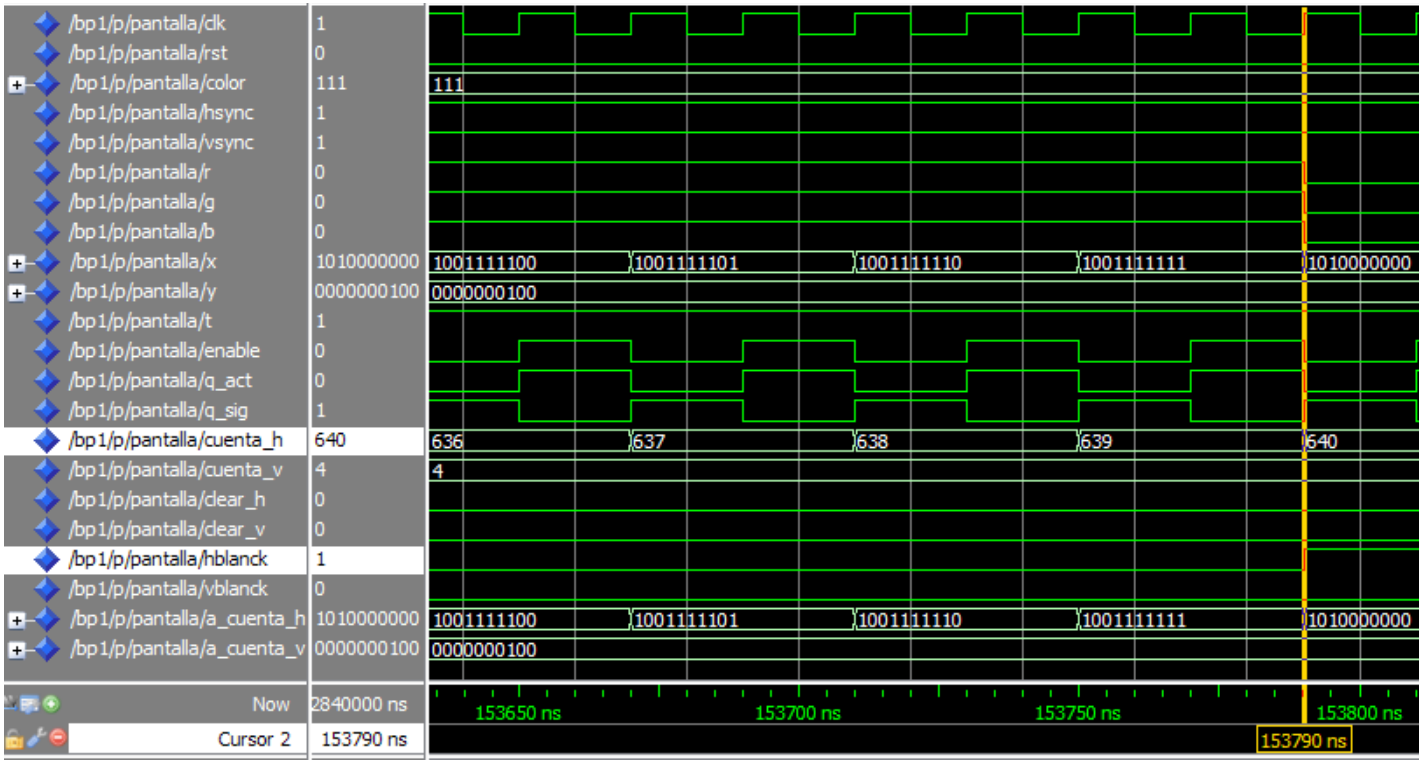


Figura 32. Simulación 10.

En esta simulación se muestra con más detalle cuando se activa la señal *HBlanck*. Se aprecia que cuando la señal *cuenta_h* (encargada de contar los píxeles cuyo período es de 40 ns) llega al píxel 640, *HBlanck* se pone a *'1'*. De esta forma se justifican los cálculos hechos en la simulación anterior [ver Figura 31].

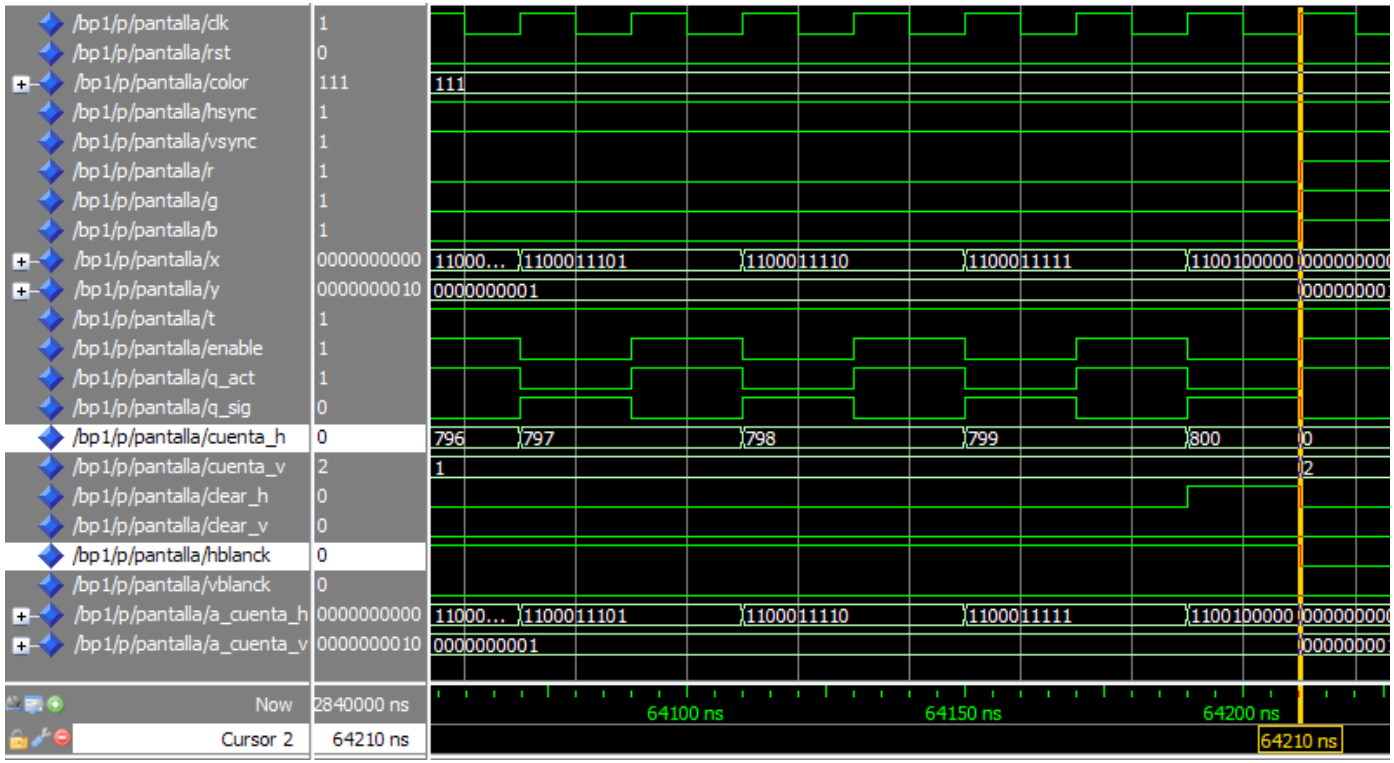


Figura 33. Simulación 11.

En esta simulación se muestra con más detalle cuando se desactiva la señal *HBlanck*. Se aprecia que cuando la señal *cuenta_h* (encargada de contar los píxeles cuyo período es de 40 ns) termina el píxel 800, *HBlanck* se pone a *'0'*. De esta forma se justifican los cálculos hechos en la simulación anterior [ver Figura 31].

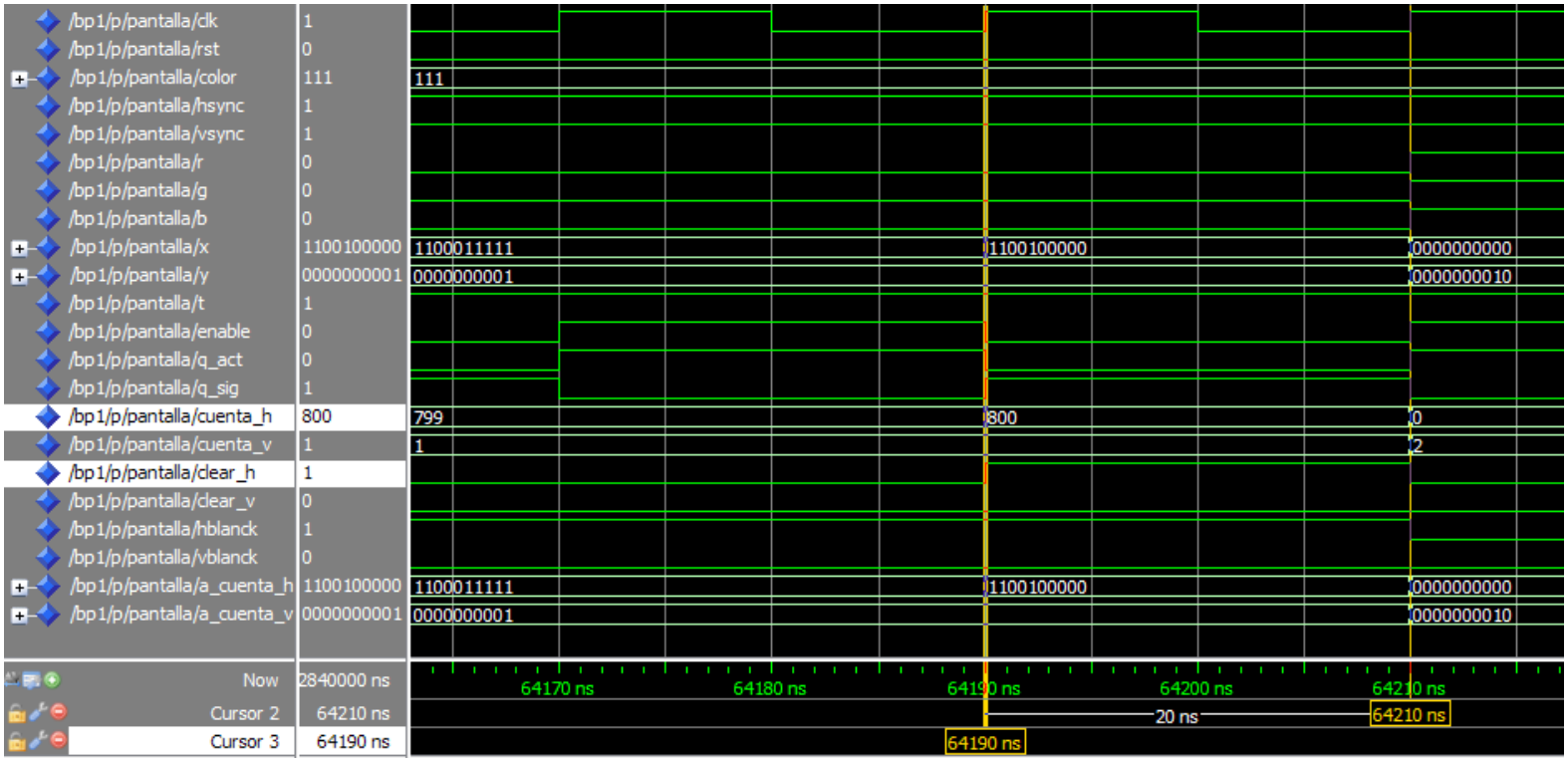


Figura 34. Simulación 12.

Esta simulación demuestra que el diseño cumple con la temporización de la señal *clear_h* para reiniciar el contador de los píxeles con período de 40 ns (reiniciar la cuenta de la señal *cuenta_h*). Esta señal es activa a nivel alto *'1'* y se aprecia como durante unos 20 ns permanece a nivel alto. Según la Tabla 14 esta señal debe activarse en el píxel 800 con un período de 40 ns, pero como el reloj interno de la FPGA utilizada en el proyecto tiene un período de 20 ns provoca que el píxel 800 de *cuenta_h* no dure los 40 ns (1*40 ns=40 ns) que debería. Esta es la explicación por la cual faltan unos 20 ns en la simulación 9 [ver Figura 31].

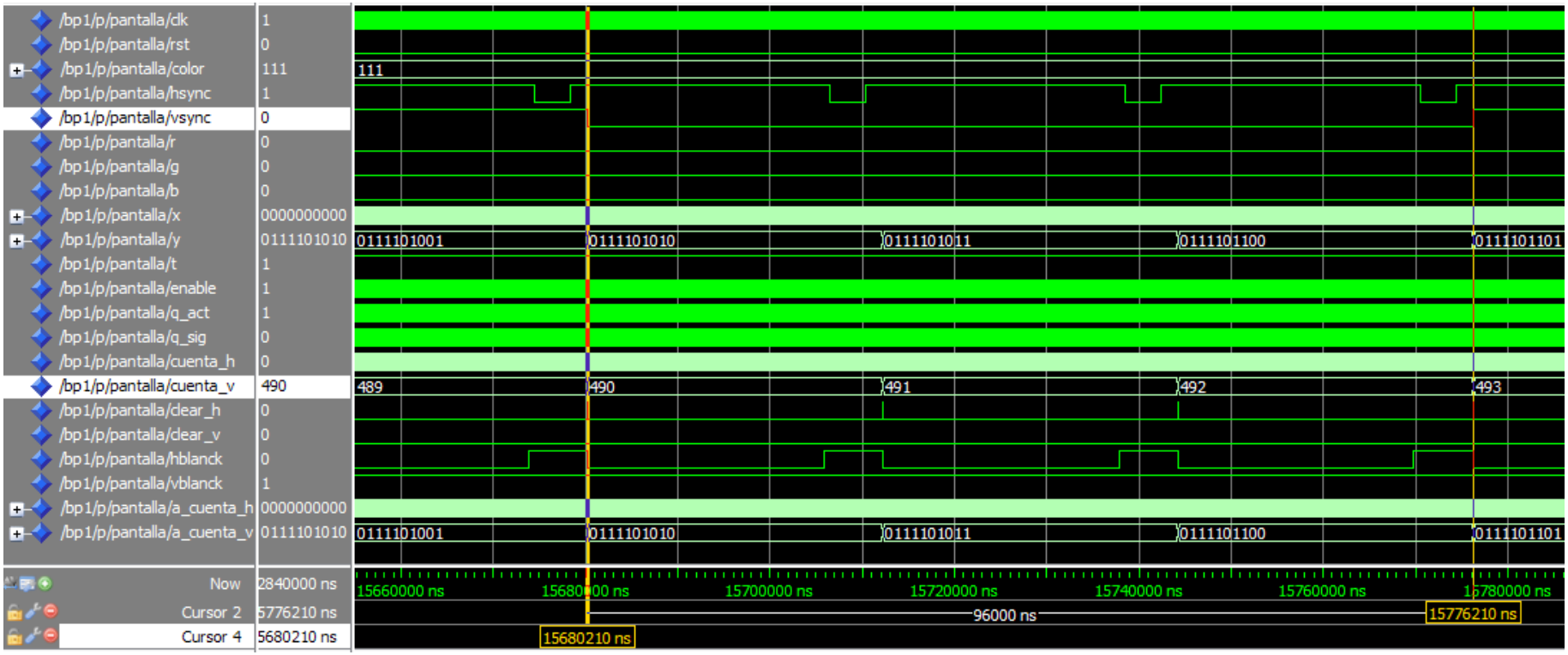


Figura 35. Simulación 13.

Esta simulación demuestra que el diseño cumple con la temporización de la señal de sincronización vertical *VSync*. Esta señal es activa a nivel bajo *'0'* y se aprecia como durante unos 96000 ns permanece a nivel bajo. Según la Tabla 14 esta señal debe activarse en la línea horizontal 490 (contabilizada a través de la señal *cuenta_v*) y mantenerse hasta terminar la línea horizontal 492 (contabilizada a través de la señal *cuenta_v*), es decir, tiene que estar activada durante 3 líneas horizontales (493-490=3). Si el diseño trabaja con un período de 40 ns y además se sabe que una línea horizontal está formada por 800 píxeles se obtienen estos 96000 ns (3*40 ns*800=96000 ns).

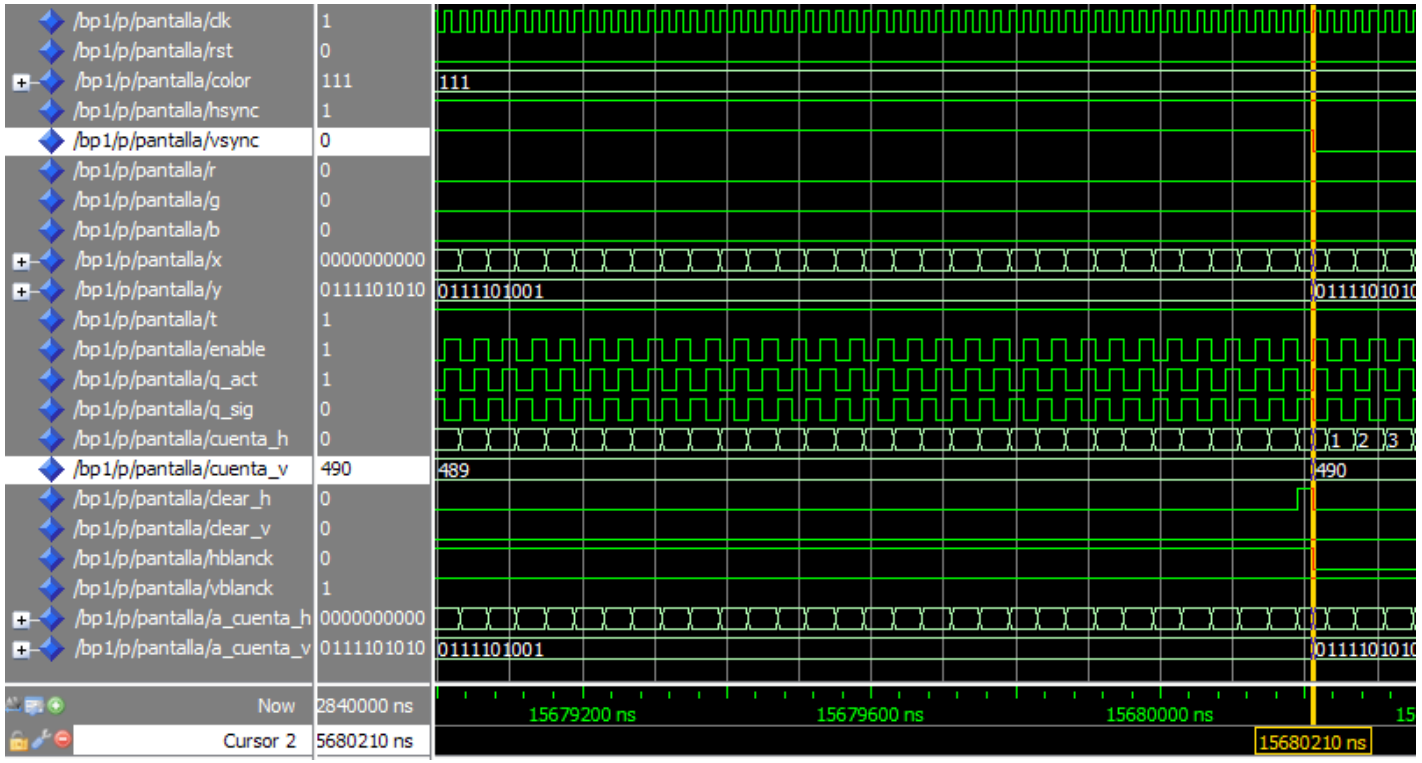


Figura 36. Simulación 14.

En esta simulación se muestra con más detalle cuando se activa la señal *VSync*. Se aprecia que cuando la señal *cuenta_v* (encargada de contar las líneas horizontales) llega a la línea 490, *VSync* se pone a *'0'*. De esta forma se justifican los cálculos hechos en la simulación anterior [ver Figura 35].

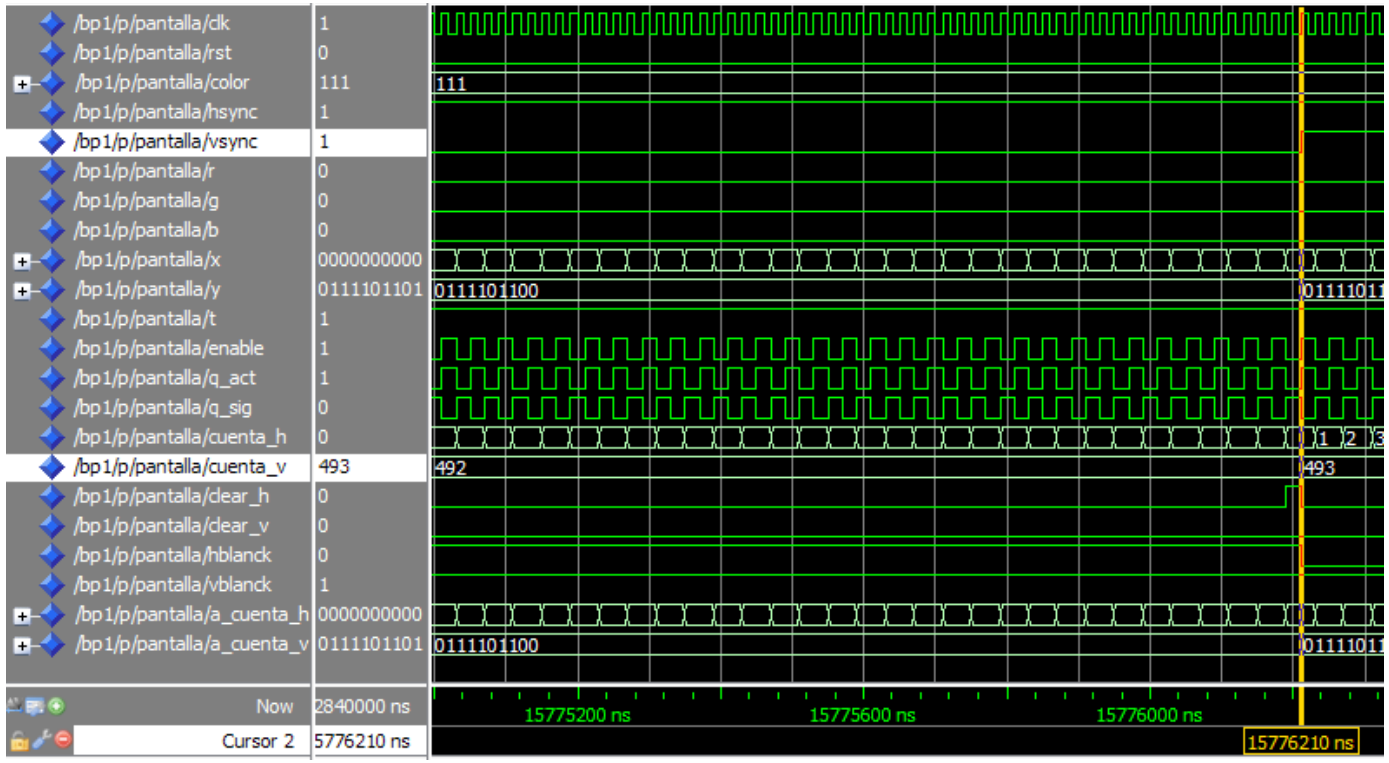


Figura 37. Simulación 15.

En esta simulación se muestra con más detalle cuando se desactiva la señal *VSync*. Se aprecia que cuando la señal *cuenta_v* (encargada de contar las líneas horizontales) termina la línea 492, *VSync* se pone a *'1'*. De esta forma se justifican los cálculos hechos en la simulación anterior [ver Figura 35].

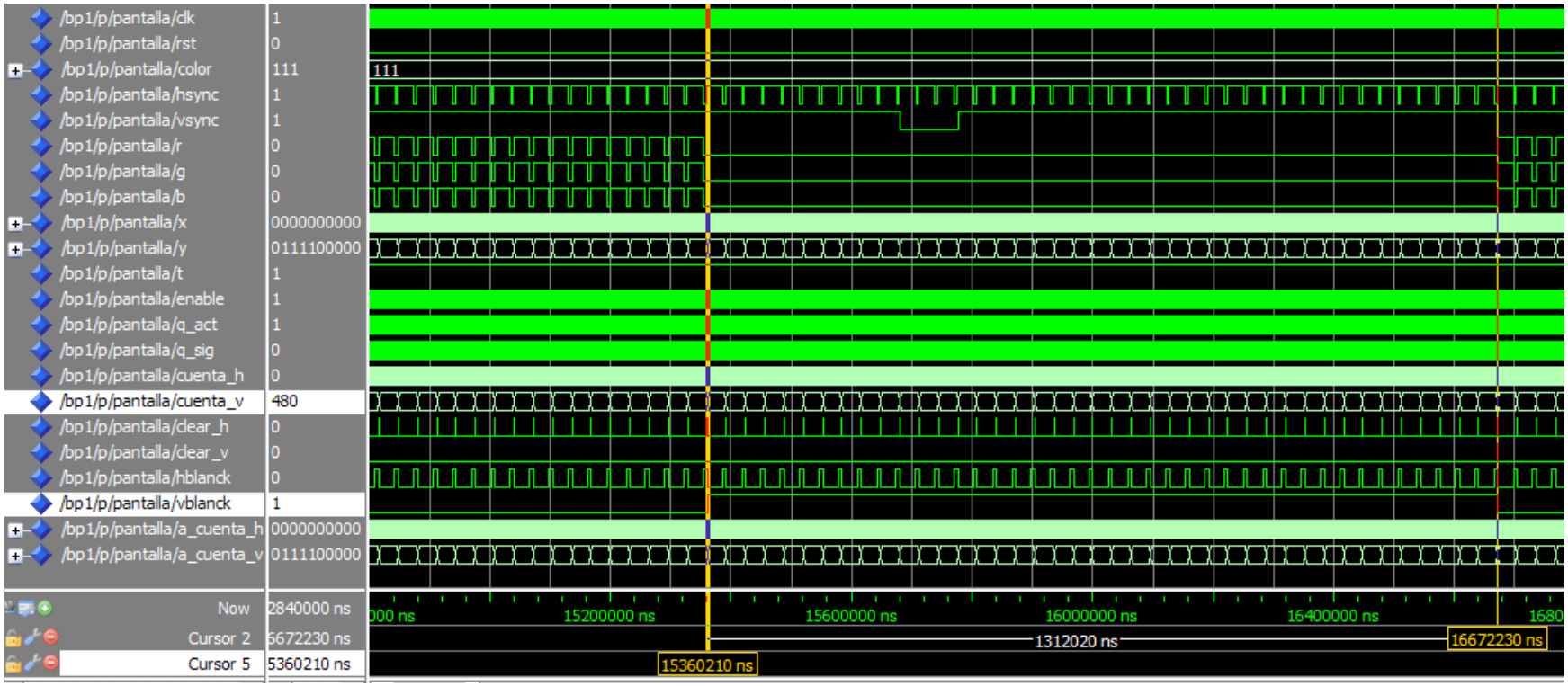


Figura 38. Simulación 16.

Con esta simulación se demuestra que el diseño cumple con la temporización de la señal de blanqueo vertical *VBlanck*. Esta señal es activa a nivel alto *'1'* y se aprecia como durante unos 1312020 ns permanece a nivel alto. Según la Tabla 14 esta señal debe activarse en la línea horizontal 480 (contabilizada a través de la señal *cuenta_v*) y mantenerse hasta terminar la línea horizontal 521 (contabilizada a través de la señal *cuenta_v*), es decir, tiene que estar activada durante 42 líneas horizontales (522-480=42). Si el diseño trabaja con un período de 40 ns y además se sabe que una línea horizontal está formada por 800 píxeles se obtienen unos 1344000 ns (42*40 ns*800=1344000 ns). Vemos que en la simulación faltan unos 31980 ns (1344000 ns-1312020 ns=31980 ns) y esto se debe a que la línea 521 de la señal *cuenta_v* se interrumpe al pasar 20 ns porque la señal *clear_v* se activa cuando se llega a ese punto y se utiliza para reiniciar el contador de *cuenta_v* [ver Figura 41].

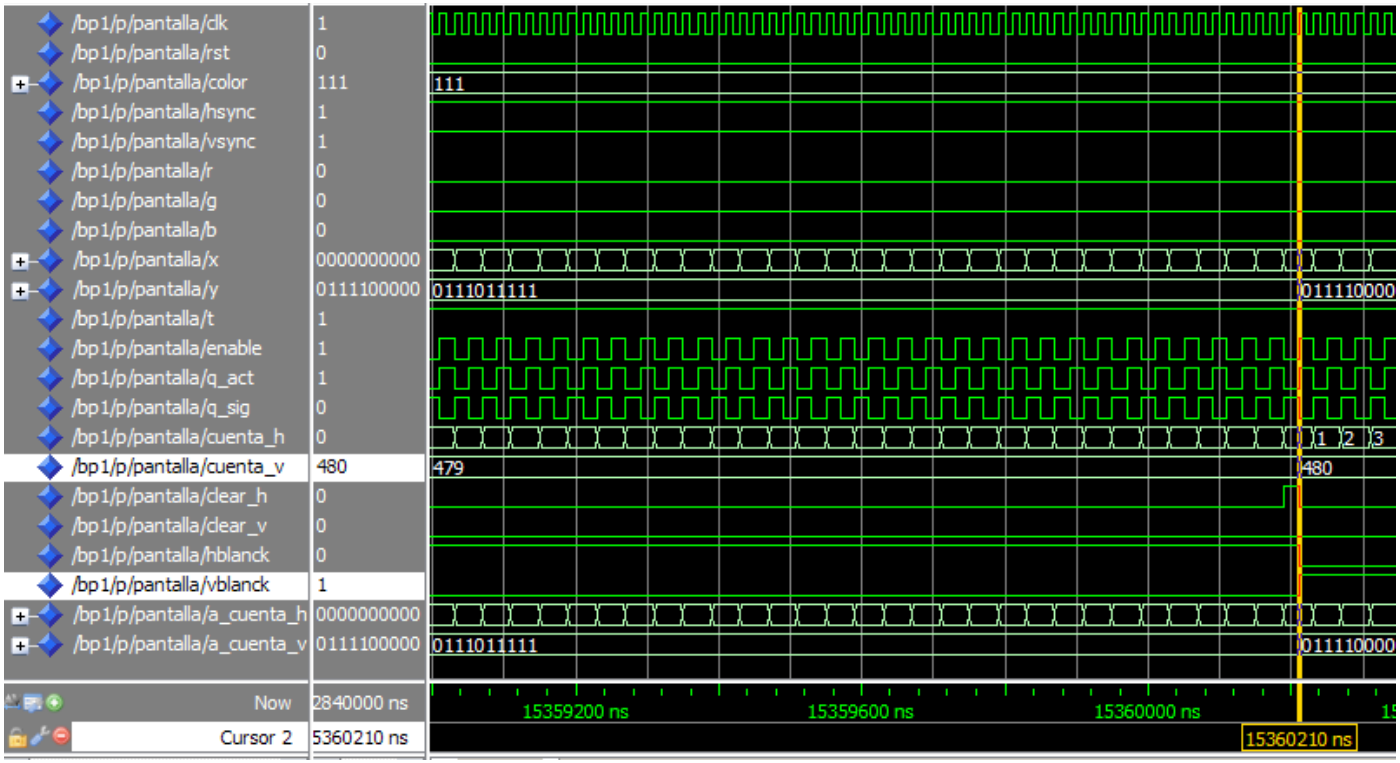


Figura 39. Simulación 17.

En esta simulación se muestra con más detalle cuando se activa la señal *VBlanck*. Se aprecia que cuando la señal *cuenta_v* (encargada de contar las líneas horizontales) llega a la línea 480, *VBlanck* se pone a *'1'*. De esta forma se justifican los cálculos hechos en la simulación anterior [ver Figura 38].

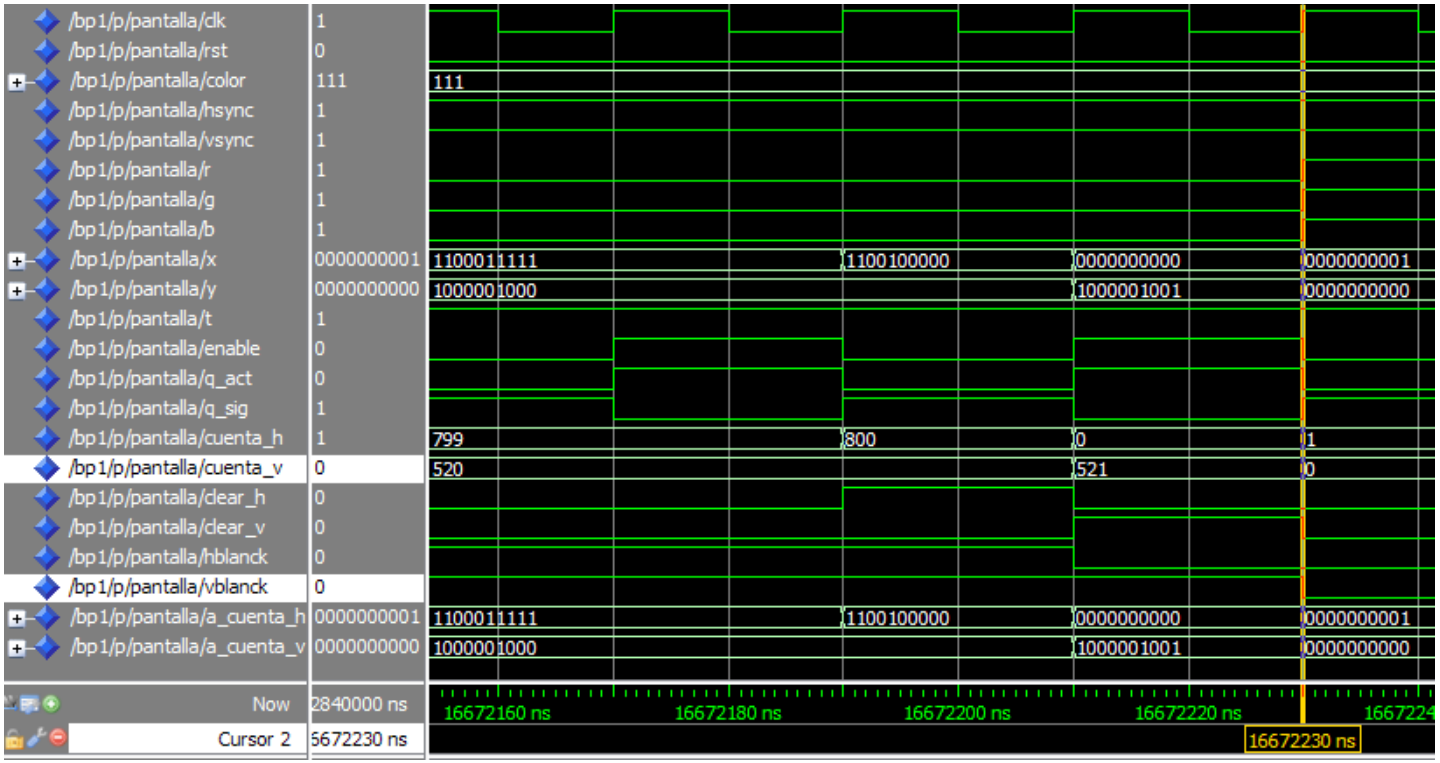


Figura 40. Simulación 18.

En esta simulación se muestra con más detalle cuando se desactiva la señal *VBlanck*. Se aprecia que cuando la señal *cuenta_v* (encargada de contar las líneas horizontales) termina la línea 521, *VBlanck* se pone a *'0'*. De esta forma se justifican los cálculos hechos en la simulación anterior [ver Figura 38].

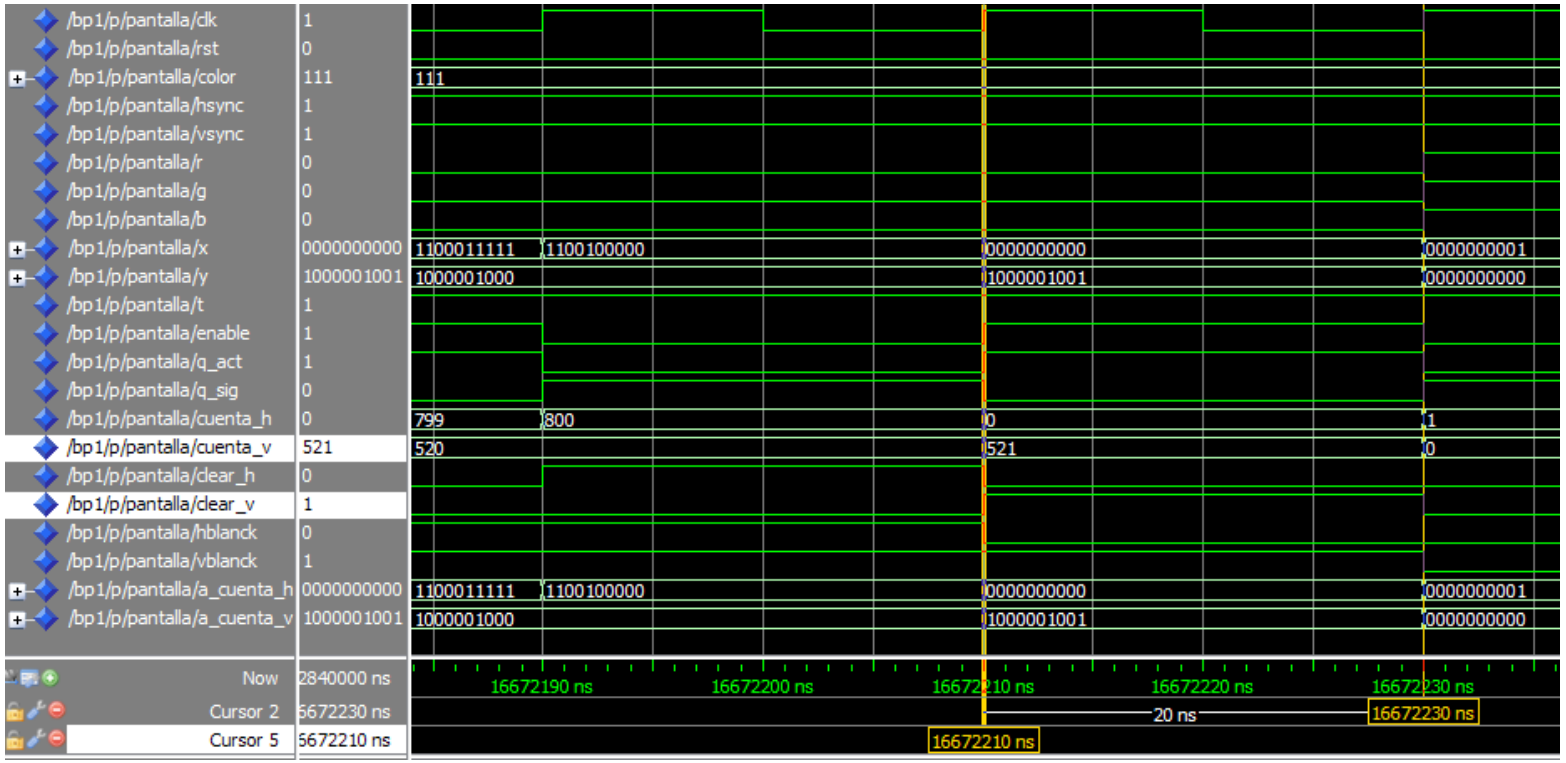


Figura 41. Simulación 19.

Esta simulación demuestra que el diseño cumple con la temporización de la señal *clear_v* para reiniciar el contador de las líneas horizontales (reiniciar la cuenta de la señal *cuenta_v*). Esta señal es activa a nivel alto *'1'* y se aprecia como durante unos 20 ns permanece a nivel alto. Según la Tabla 14 esta señal debe activarse en la línea 521, pero como el reloj interno de la FPGA utilizada en el proyecto tiene un período de 20 ns provoca que esta línea 521 de *cuenta_v* no dure los 32000 ns (800*40 ns=32000 ns) que debería. Esta es la explicación por la cual faltan unos 31980 ns en la simulación 16 [ver Figura 38].

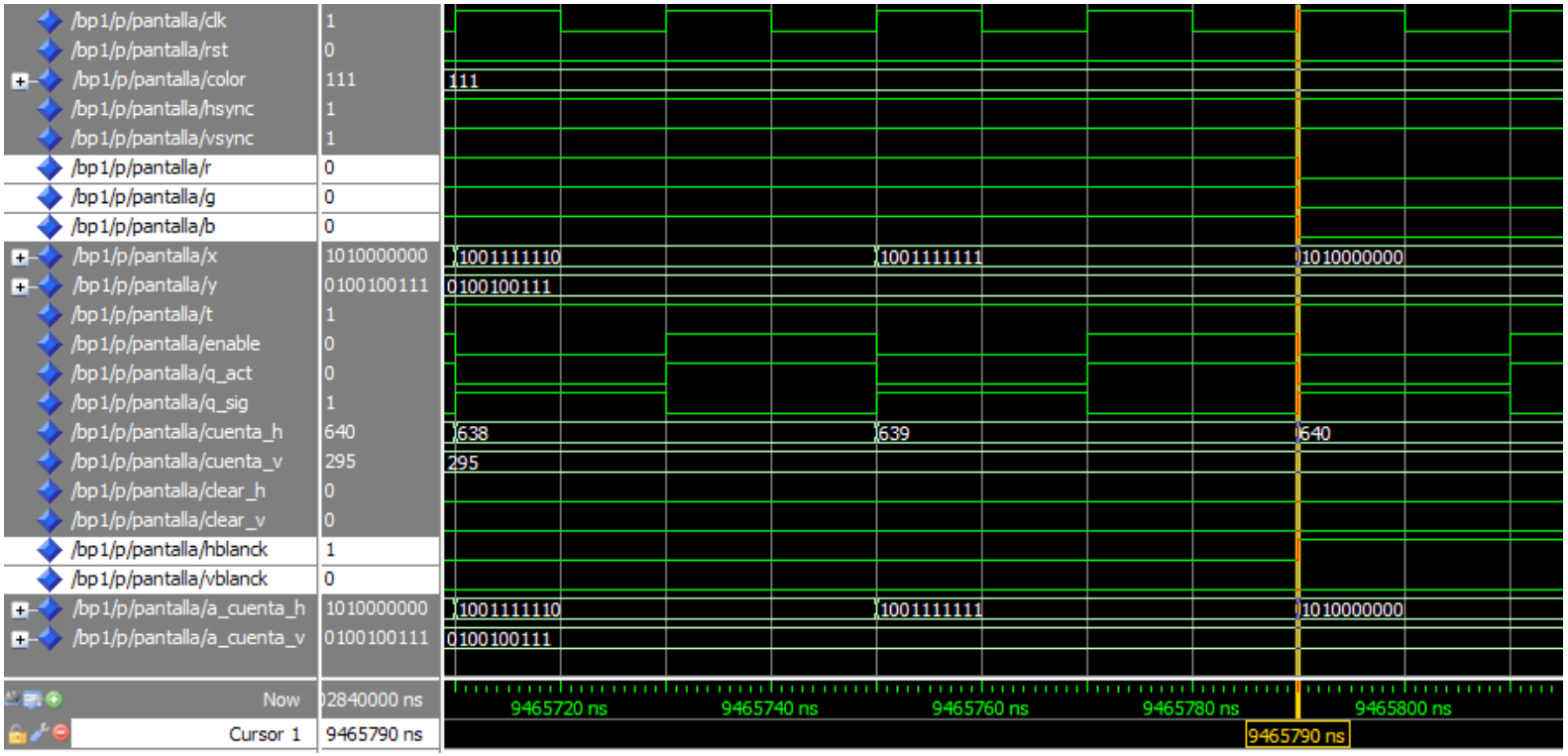


Figura 42. Simulación 20.

Con esta simulación se demuestra que el bloque blanqueador funciona adecuadamente, ya que pone a *'0'* las señales *R*, *G* y *B* respectivamente (color negro según la Tabla 12) cuando se activa alguna de las dos señales *HBlanck* o *VBlanck*. En este caso se aprecia como la señal que se activa es *HBlanck*.

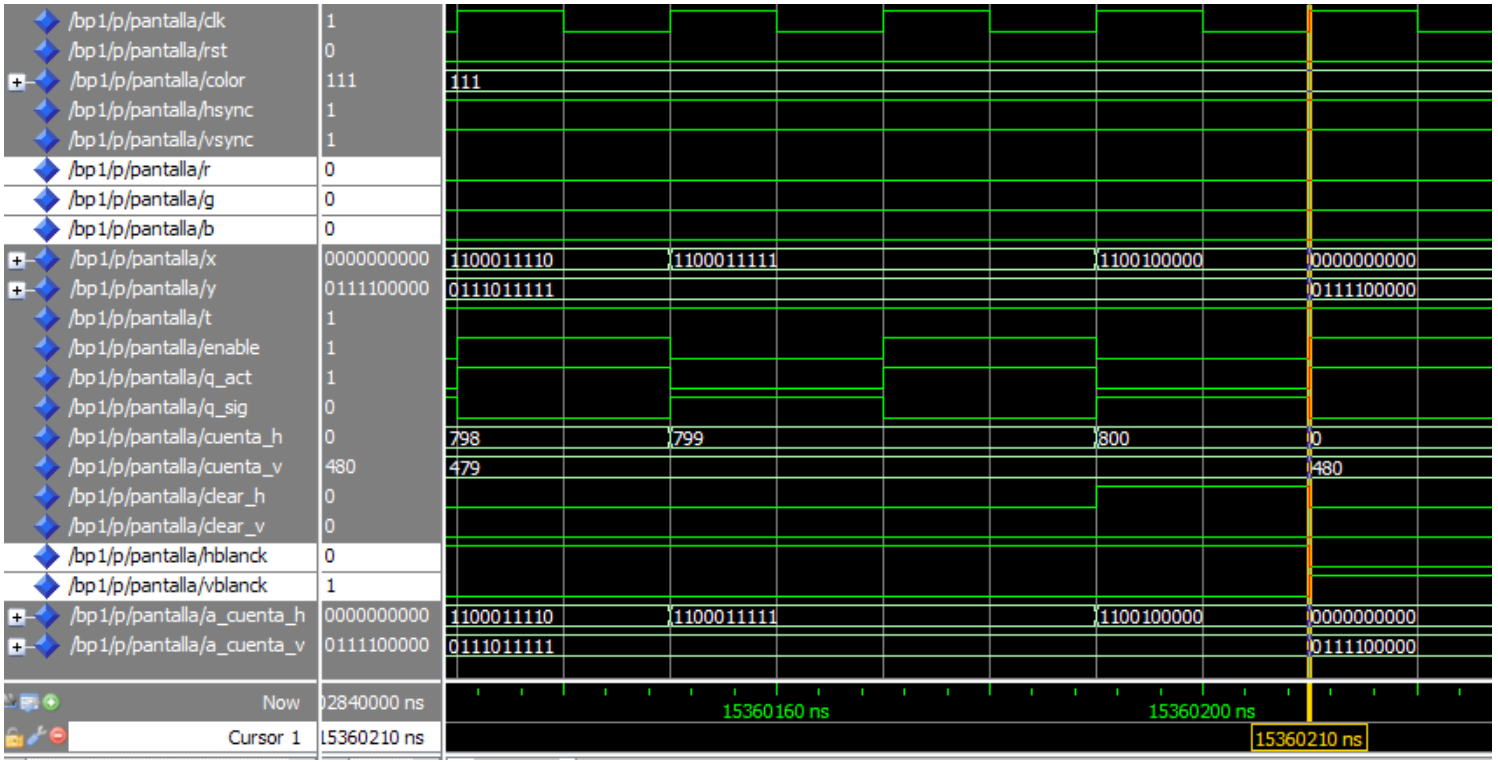


Figura 43. Simulación 21.

Con esta simulación se demuestra que el bloque blanqueador funciona adecuadamente, ya que pone a *'0'* las señales *R*, *G* y *B* respectivamente (color negro según la Tabla 12) cuando se activa alguna de las dos señales *HBlanck* o *VBlanck*. En este caso se aprecia como la señal que se activa es *VBlanck*.

11. Bloque 3: Controlador y dibujador [DRIVER].

11.1. Descripción del problema.

En primer lugar este tercer bloque se ha llamado Controlador o dibujador debido a que es el encargado de controlar la conexión o comunicación entre los bloques anteriores diseñados (1 y 2). En este punto de conexión entre el protocolo *PS/2* y el *VGA* es imprescindible controlar con mucha exactitud varios datos. Principalmente la información que proviene del teclado a través del puerto *PS/2*, para luego identificar que es dicho dato. Seguidamente y también muy importante es conocer la localización exacta del píxel que se está mostrando en la pantalla en cada momento, para así poder establecer que color se deberá dibujar en ese píxel dependiendo de la información recibida del teclado. Todo lo expuesto anteriormente influye directamente en el diseño del controlador, como también influye la correcta utilización de las características que nos brinda la *Spartan 3E-Starter Board*, ya que la complejidad de este bloque aumenta con respecto al de los dos anteriores. Por este motivo es primordial conocer lo mejor posible los tiempos y formas de onda de los bloques de teclado y de pantalla.

11.2. Diseño, tipografía y códigos de escaneo: Set 2.

El llamado Controlador o dibujador o bloque 3, funciona utilizando la trama de 8 bits recibida desde el teclado, la interpreta y genera nuevos valores que envía a su vez a la pantalla o monitor. Toda esta sucesión de acciones tiene una explicación más detallada y en este apartado se pretende aclarar todas las cuestiones relacionadas con este bloque.

Como se ha mencionado anteriormente, estamos usando una resolución de 640x480 píxeles, es decir, los píxeles que vemos en pantalla son 640 de ancho y 480 de alto. A partir de estos datos se planteó la idea de dividir el monitor en sectores cuadrados de 20x20 píxeles; serían cuadros pequeños que de ancho tendrían 20 píxeles y otros 20 de alto. Esta organización de la pantalla se hace primero que todo con el objetivo de crear una especie de matriz virtual; de esta forma quedaría una distribución de 32 columnas por 24 filas, logrando que el acceso a los píxeles de la pantalla que se quieran dibujar sea de una forma más fácil e intuitiva [ver Figura 44].

El hecho de que el control de la posición de los pixeles a la hora de dibujar sea más clara y sencilla, permite en segundo lugar que el código de síntesis de hardware sea mucho más compacto y menos extenso, básicamente reduce bastante la síntesis del código. En tercer lugar la repartición de pantalla en bloques cuadrados, posibilita que la gestión de colores sea también mucho más simple; esto se debe a que dibujando una sección de 20x20 píxeles de un mismo color se abarca mucha más área de la pantalla, exactamente 400 pixeles por cuadro definido. Por último y no menos importante, el poder acceder a cada parte del monitor mediante estos sectores predefinidos, hace que el diseño de la tipografía a presentar por pantalla se pueda preestablecer con antelación y saber con exactitud qué sectores hay que dibujar en cada momento, con qué color y para cada carácter específico.

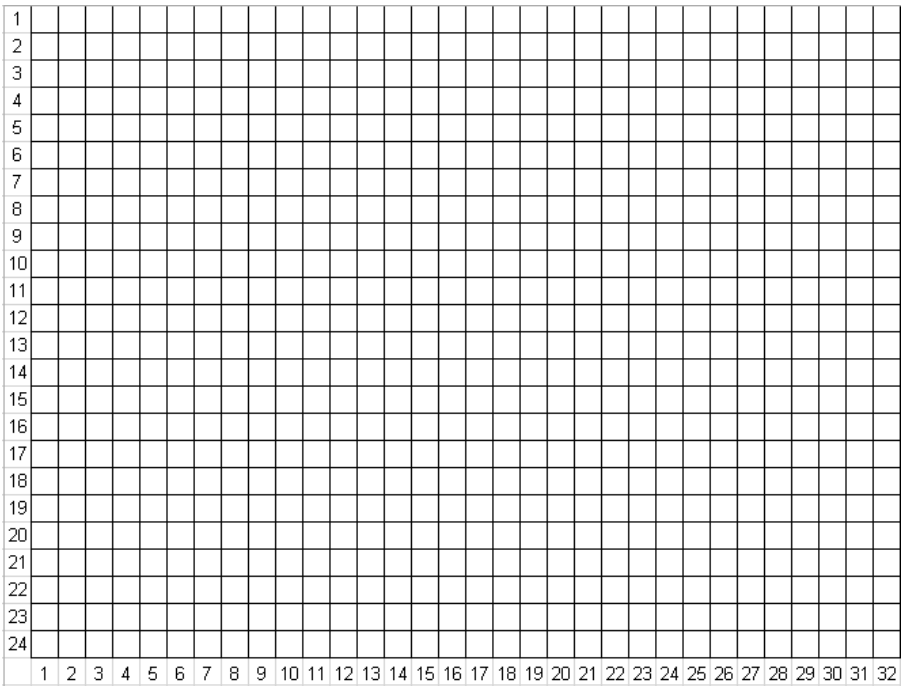


Figura 44. Distribución en sectores de 20x20 pixeles de la pantalla.

En este trabajo para el diseño de la tipografía que se mostrará en pantalla, sólo se usaron tres colores (blanco, negro y rojo). El estilo de los caracteres que se diseñaron es muy básico y para nada vistoso; principalmente esto se debe a que cada carácter se conforma dibujando sectores de 20x20 píxeles en color negro. Esta razón pone de manifiesto que las terminaciones y el acabado de los caracteres sean cuadrados y con aristas rectas. La presentación en pantalla se creó de tal forma que tuviera gran contraste y que sobre todo se resaltara el carácter a mostrar, que es realmente lo importante en este proyecto. Esto se logró utilizando un fondo de pantalla en blanco, los caracteres con un tamaño considerable en color negro y todo el diseño se centró a partir del sector (columna *12* y fila *16*); además de centrar todo el diseño, cada carácter se mostrará en el interior de un marco de color rojo para lograr un mayor enfoque de estos [véase Figura 45].

Este marco rojo, está compuesto por dos fragmentos de filas de sectores (filas *7* y *17*, fragmentos que ocupan sectores de las columnas *11* a la *21*) y otros dos fragmentos de columnas de sectores (columnas *11* y *21*, fragmentos que ocupan sectores de las filas *7* a la *17*). Por tanto, el área delimitada por el marco es de 9x9 sectores de 20x20 píxeles cada uno y cada carácter representado dentro del recuadro rojo ocupará una superficie de 5x7 cuadros de 20x20 píxeles, todos centrados como se ha explicado antes.

La idea de funcionamiento que se estableció es la que una vez se reciban los 8 bits del *Scancode* de cada carácter en específico o trama de bits, estos se analizan y se determina que carácter estamos recibiendo. Una vez reconocido el carácter se establecen los colores que hay que enviar al monitor, junto con los sectores que hay que dibujar del mismo. Cuando se dibuje una letra en pantalla, esta se mantendrá visualizándose mientras no se presione otra tecla en el teclado. De los tres posibles conjuntos de *Scancodes* que existen, usaremos el *Set Scancode 2* [ver Tabla 19].

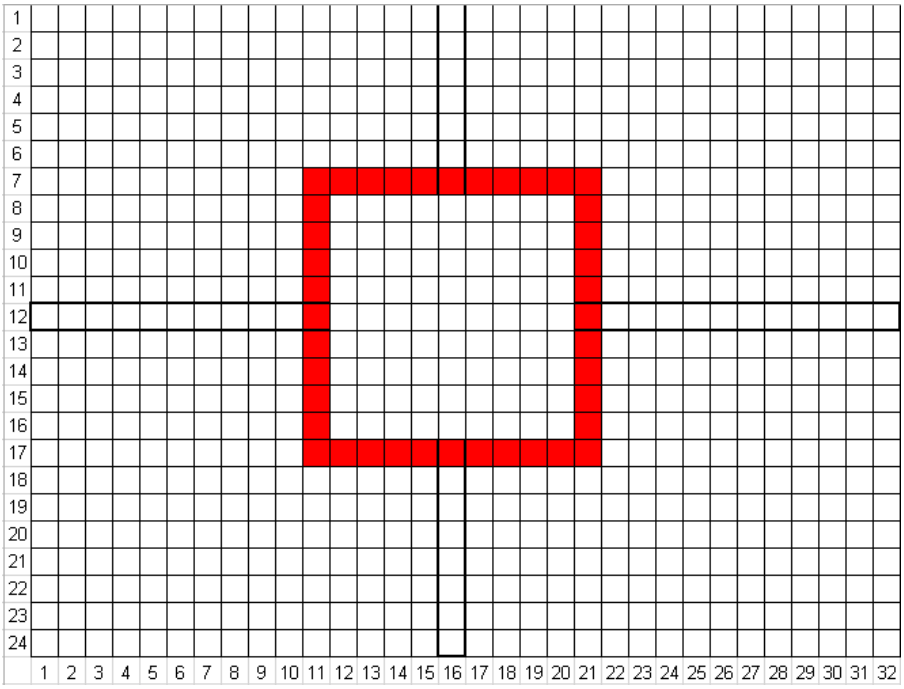
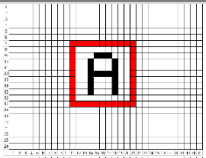
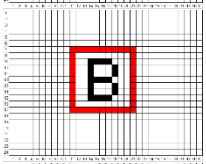
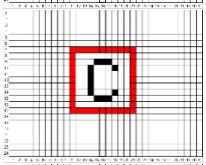
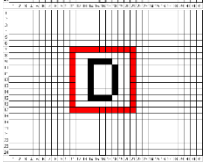
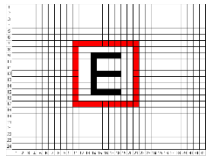
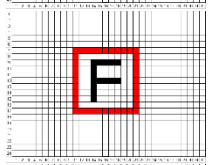
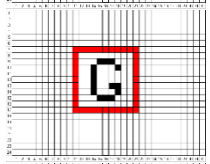
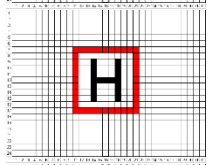
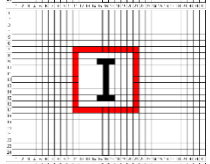
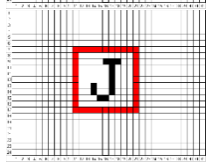
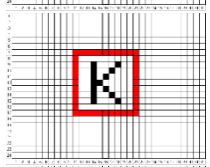
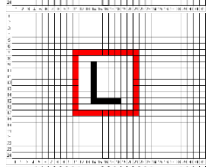
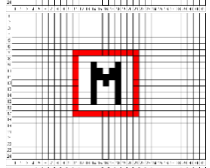
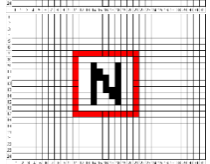
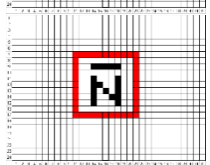
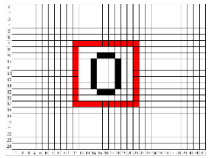
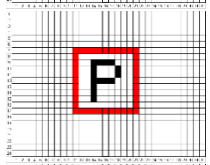
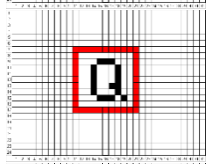
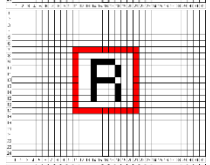
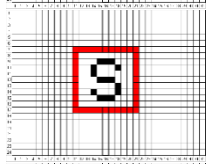
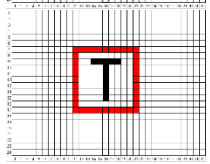
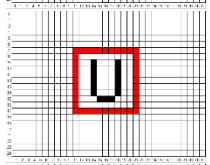
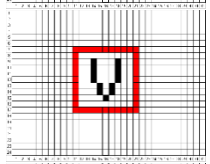
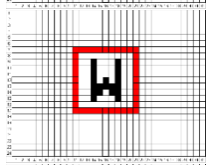
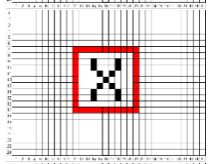
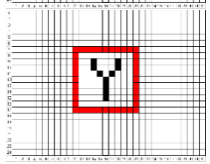


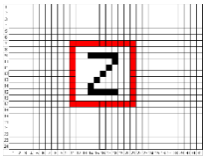
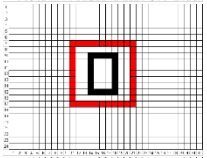
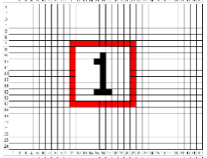
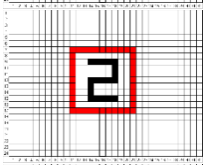
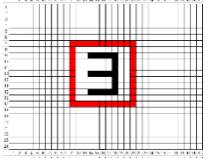
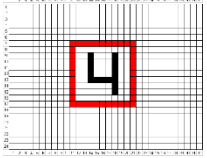
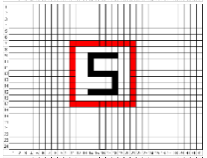
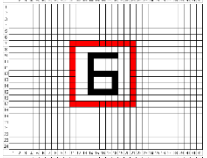
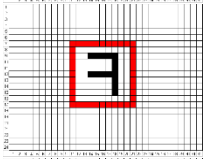
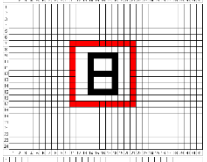
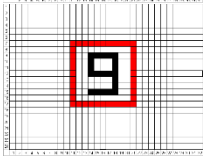
Figura 45. Marco de presentación de los caracteres del teclado.

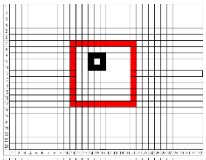
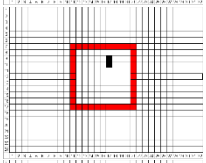
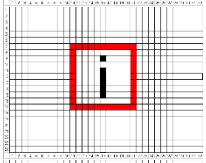
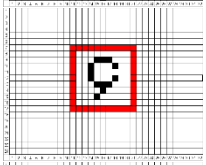
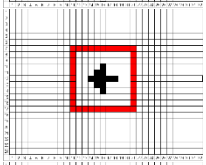
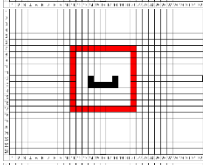
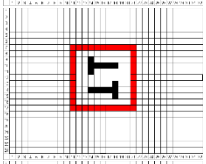
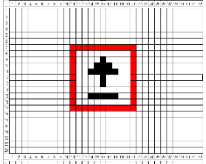
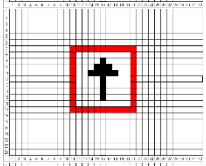
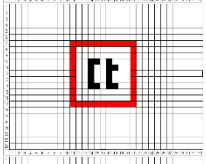
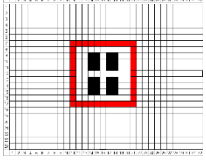
En la Tabla 19 se presenta la totalidad de caracteres implementados en este proyecto. Se muestran por cada carácter su *Makecode*, su correspondiente equivalente binario (que es con lo que podemos trabajar a nivel lógico), también su *Breakcode* (nótese que para todos los caracteres es el mismo, debido a que no se contemplan las segundas funciones de las teclas, ni se implementó la comunicación desde la *FPGA* al teclado), además su correspondiente código binario y por último la imagen de la tipografía establecida para cada carácter (es lo que se visualizará por pantalla).

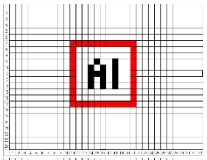
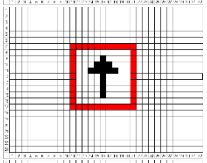
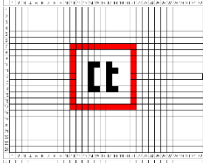
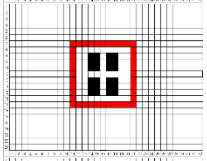
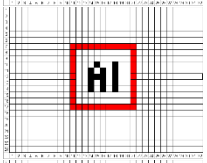
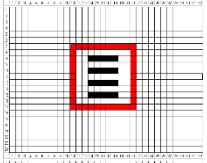
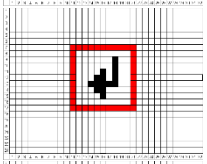
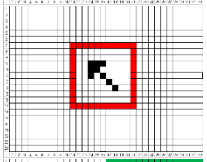
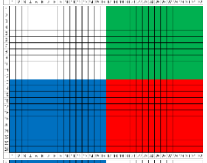
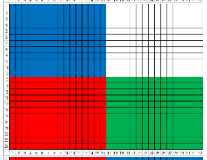
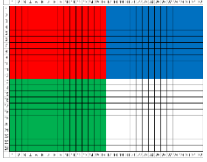
Carácter	*Makecode*	Binario	*Breakcode*	Binario	Imagen
A	1C	00011100	F0	11110000	
B	32	00110010	F0	11110000	
C	21	00100001	F0	11110000	
D	23	00100011	F0	11110000	

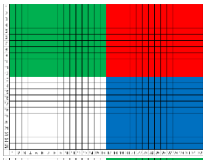
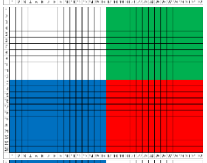
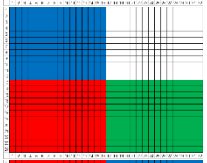
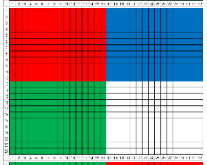
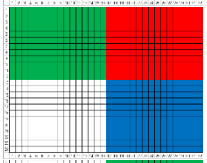
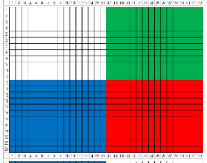
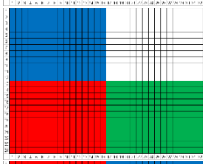
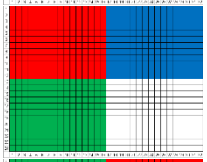
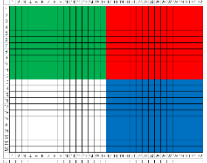
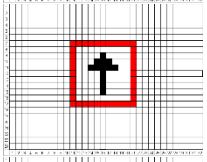
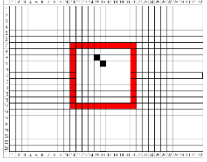
E	24	00100100	F0	11110000	
F	2B	00101011	F0	11110000	
G	34	00110100	F0	11110000	
H	33	00110011	F0	11110000	
I	43	01000011	F0	11110000	
J	3B	00111011	F0	11110000	
K	42	01000010	F0	11110000	
L	4B	01001011	F0	11110000	
M	3A	00111010	F0	11110000	
N	31	00110001	F0	11110000	
Ñ	4C	01001100	F0	11110000	

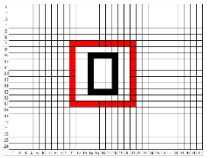
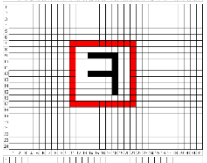
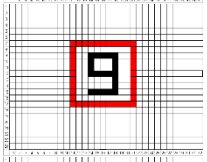
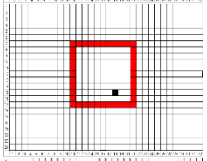
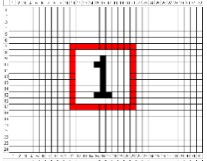
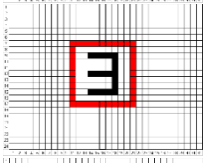
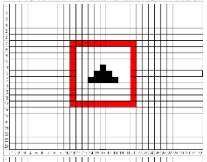
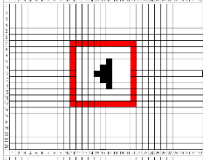
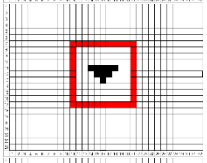
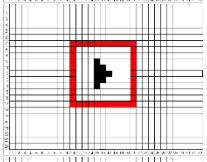
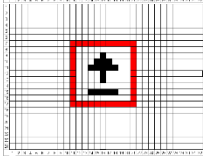
O	44	01000100	F0	11110000	
P	4D	01001101	F0	11110000	
Q	15	00010101	F0	11110000	
R	2D	00101101	F0	11110000	
S	1B	00011011	F0	11110000	
T	2C	00101100	F0	11110000	
U	3C	00111100	F0	11110000	
V	2A	00101010	F0	11110000	
W	1D	00011101	F0	11110000	
X	22	00100010	F0	11110000	
Y	35	00110101	F0	11110000	

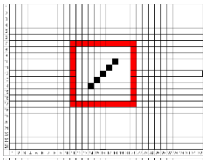
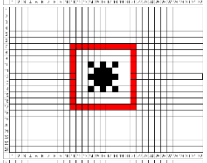
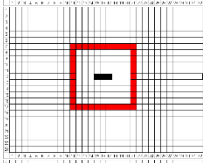
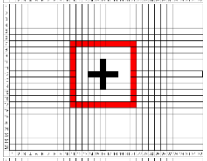
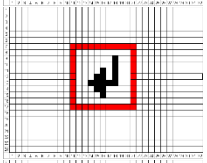
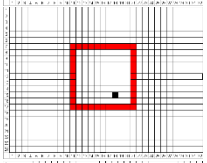
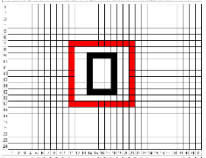
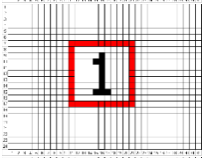
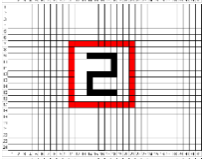
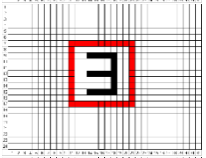
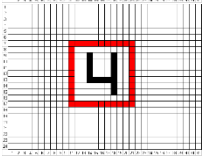
Z	1A	00011010	F0	11110000	
0	45	01000101	F0	11110000	
1	16	00010110	F0	11110000	
2	1E	00011110	F0	11110000	
3	26	00100110	F0	11110000	
4	25	00100101	F0	11110000	
5	2E	00101110	F0	11110000	
6	36	00110110	F0	11110000	
7	3D	00111101	F0	11110000	
8	3E	00111110	F0	11110000	
9	46	01000110	F0	11110000	

º	0E	00001110	F0	11110000	
'	4E	01001110	F0	11110000	
¡	55	01010101	F0	11110000	
ç	5D	01011101	F0	11110000	
Delete	66	01100110	F0	11110000	
Espacio	29	00101001	F0	11110000	
Tabulación	0D	00001101	F0	11110000	
Bloq. Mayus	58	01011000	F0	11110000	
Shift Izq.	12	00010010	F0	11110000	
Control Izq.	14	00010100	F0	11110000	
Windows Izq.	1F	00011111	F0	11110000	

Alt Izq.	11	00010001	F0	11110000	
Shift Der.	59	01011001	F0	11110000	
Control Der.	14	00010100	F0	11110000	
Windows Der.	27	00100111	F0	11110000	
Alt Der.	11	00010001	F0	11110000	
Aplicaciones	2F	00101111	F0	11110000	
Enter	5A	01011010	F0	11110000	
Esc	76	01110110	F0	11110000	
F1	5	00000101	F0	11110000	
F2	6	00000110	F0	11110000	
F3	4	00000100	F0	11110000	

F4	0C	00001100	F0	11110000	
F5	3	00000011	F0	11110000	
F6	0B	00001011	F0	11110000	
F7	83	10000011	F0	11110000	
F8	0A	00001010	F0	11110000	
F9	1	00000001	F0	11110000	
F10	9	00001001	F0	11110000	
F11	78	01111000	F0	11110000	
F12	7	00000111	F0	11110000	
Imp. Pant.	12	00010010	F0	11110000	
`	54	01010100	F0	11110000	

Insertar	70	01110000	F0	11110000	
Inicio	6C	01101100	F0	11110000	
Re. Pág.	7D	01111101	F0	11110000	
Suprimir	71	01110001	F0	11110000	
Fin	69	01101001	F0	11110000	
Av. Pág.	7A	01111010	F0	11110000	
Flecha Arriba	75	01110101	F0	11110000	
Flecha Izquierda	6B	01101011	F0	11110000	
Flecha Abajo	72	01110010	F0	11110000	
Flecha Derecha	74	01110100	F0	11110000	
Bloq. Num.	77	01110111	F0	11110000	

KP_/	4A	01001010	F0	11110000	
KP_*	7C	01111100	F0	11110000	
KP_-	7B	01111011	F0	11110000	
KP_+	79	01111001	F0	11110000	
KP_Enter	5A	01011010	F0	11110000	
KP_.	71	01110001	F0	11110000	
KP_0	70	01110000	F0	11110000	
KP_1	69	01101001	F0	11110000	
KP_2	72	01110010	F0	11110000	
KP_3	7A	01111010	F0	11110000	
KP_4	6B	01101011	F0	11110000	

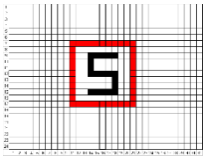
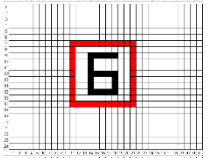
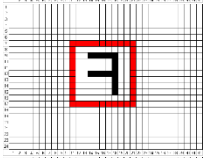
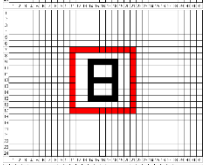
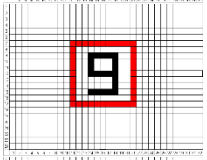
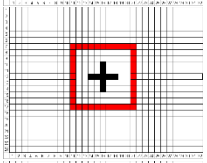
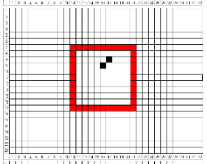
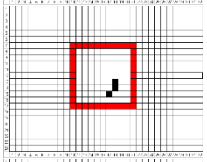
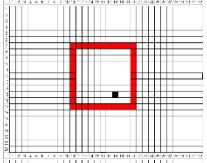
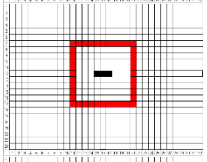
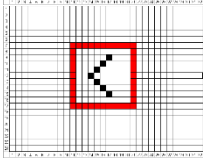
KP_5	73	01110011	F0	11110000	
KP_6	74	01110100	F0	11110000	
KP_7	6C	01101100	F0	11110000	
KP_8	75	01110101	F0	11110000	
KP_9	7D	01111101	F0	11110000	
+	5B	01011011	F0	11110000	
´	52	01010010	F0	11110000	
,	41	01000001	F0	11110000	
.	49	01001001	F0	11110000	
-	4A	01001010	F0	11110000	
<	61	01100001	F0	11110000	

Tabla 19. Set 2 de los Códigos de Escaneo o *Set 2 Scancodes*.

Si se analiza con detenimiento la Tabla 19, se puede notar que hay ciertos pares de caracteres que coinciden. Esto es porque al no implementarse las segundas funciones de teclas, los *Breakcodes* coinciden al igual que los *Makecodes*. Por lo tanto, al presionar cualquiera de las dos teclas el resultado en pantalla es el mismo. Esto sucede con los pares siguientes: *Shift Izquierdo-Imp. Pant.*, *Control Izquierdo-Control Derecho*, *Alt Izquierdo-Alt Derecho*, *Enter-KP_Enter*, *Insert-KP_0*, *Inicio-KP_7*, *Re. Pág.-KP_9*, *Suprimir-KP_.*, *Fin-KP_1*, *Av. Pág-KP_3*, *Flecha Arriba-KP_8*, *Flecha Izquierda-KP_4*, *Flecha Abajo-Kp-2*, *Flecha Derecha-KP_6* y *KP_/-Guion*.

Es imprescindible aclarar que el funcionamiento normal de la parte numérica de un teclado es que cuando se presiona la tecla *Bloq. Num.* se activa dicho teclado numérico, es decir, los números corresponden a los indicados al igual que las operaciones matemáticas básicas. De lo contrario cuando no se presiona *Bloq. Num.* las teclas mantienen su carácter excepto el *2*, *4*, *6* y *8*, las cuales cambian por las flechas de dirección. Por este motivo, cuando presionamos dicha tecla es en el momento que coinciden los caracteres mencionados antes del *Key Pad*.

En la Tabla 19 se puede comprobar que las teclas de funciones (teclas del *F1* a la *F12*) no se han representado por sus nombres específicos. En este proyecto se implementó un patrón de pruebas para cada una de ellas, básicamente se dividió la pantalla en 4 sectores partiendo del centro y se rellenó cada sector con un color (blanco, verde, rojo y azul). A medida que se aumenta la tecla de función este patrón va rotando en el sentido de las agujas del reloj. Esta idea del patrón de pruebas surgió en un principio para ir probando la comunicación del bloque de teclado con el de la pantalla *VGA*.

11.3. Especificaciones del bloque 3.

11.3.1. Interfaz.

En este inciso se especifican cuáles son las entradas y salidas (E/S) del actual bloque 3. Las que se pueden ver en el anexo de este documento 16.3 más adelante. Las señales de este bloque son:

Entradas

- **clk**: señal que representa el reloj de la *FPGA*, el cual fija todo el diseño. Tiene una frecuencia de 50 MHz lo que produce un periodo de 20 ns.
- **rst**: señal de inicialización asíncrona del sistema, activa por nivel alto.
- **c_tecla:** señal que se transmite desde el teclado con los 8 bits de información de la tecla presionada.
- **error**: señal que proviene del bloque del teclado si existiese algún fallo en la transmisión. Esta señal se utiliza para saber cuándo dibujar toda una pantalla en color rojo, activa a nivel alto.
- **c_X:** señal de 10 bits que proviene del bloque *VGA*, la que se utiliza para poder barrer la pantalla en la dirección horizontal. La señal es de 10 bits para poder alcanzar los 640 pixeles de ancho (2^10=1024).
- **c_Y:** señal de 10 bits que proviene del bloque *VGA*, la que se utiliza para poder barrer la pantalla en la dirección vertical. La señal es de 10 bits para poder alcanzar los 480 pixeles de alto (2^10=1024).

Salidas

- **c_color:** señal que indica el código de 3 bits que designa el color a mostrar en cada píxel que se vaya a procesar.

11.3.2. Funcionalidad.

Como se ha dicho apartados anteriores este tercer bloque nombrado Controlador o dibujador, relaciona y comunica los anteriores. Es decir, el Controlador es el encargado de dibujar en pantalla los patrones prefijados para cada carácter del teclado y todo esto es en dependencia de la información que llegue desde el teclado y como la gestione e interprete el propio controlador para saber qué datos debe enviar a la pantalla.

Por tanto habrá que conocer las características que se usarán de la placa de *Xilinx* [ver Figura 46]; la más importante y la que más se tendrá en cuenta para su diseño, es su *reloj principal*; cuyo valor de frecuencia es de 50 MHz, lo que implica que tenga un período de 20 ns. El pin asignado en la placa para este reloj se muestra en la Tabla 20. La última característica a usar de la placa en este diseño, es la implementación de un *Reset* para volver al estado inicial por si hubiera algún problema o posible desajuste de funcionamiento. Este botón o interruptor fue asignado en este proyecto a un pin específico de la *FPGA*, cuya dirección viene en la Tabla 20.

Elemento	Pin
Reloj	C9
Reset	L13

Tabla 20. Pines de la *FPGA* para el bloque 3 (26).

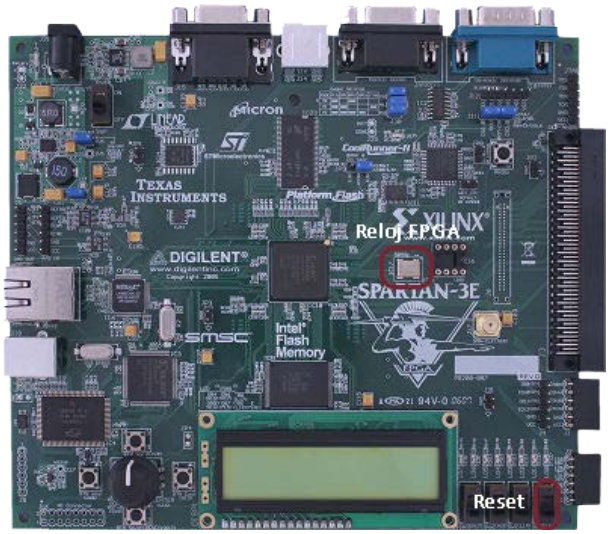


Figura 46. Características usadas de la placa *Spartan 3E-Starter Board* para el bloque 3 (26).

11.3.3. Arquitectura.

En este apartado sólo se muestran las partes o bloques del circuito, que se explicarán con detalle en el siguiente apartado 11.3.4. En la Tabla 21 se enumeran todos los bloques del circuito y en la Figura 47 se puede ver el diagrama de bloques completo de la arquitectura.

Bloque	Descripción
1	Convertidor de 10 bits de std_logic_vector a enteros para las 'X'.
2	Convertidor de 10 bits de std_logic_vector a enteros para las 'Y'.
3	Conformador de columnas.
4	Conformador de filas.
5	Selector de patrones a imprimir.

Tabla 21. Resumen de los bloques del bloque 3.

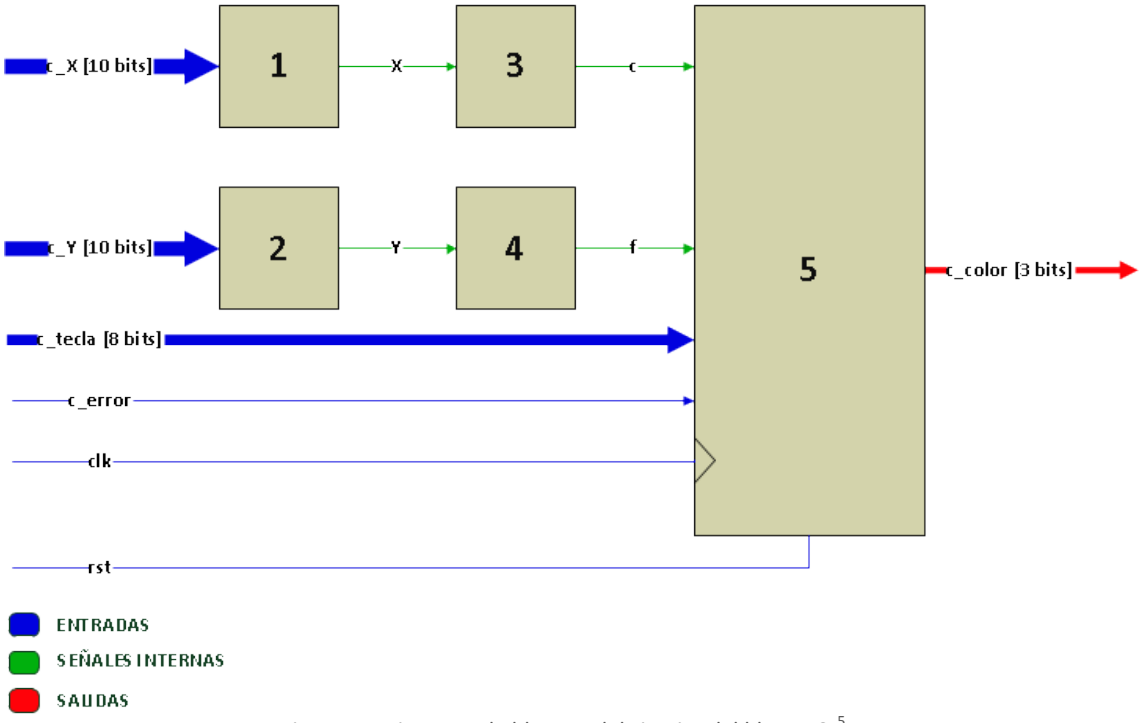


Figura 47. Diagrama de bloques del circuito del bloque 3.[5]

11.3.4. Diseño detallado y síntesis del circuito.

Según se observa en la Tabla 21, el Driver o dibujador está compuesto por un total de 5 bloques individuales, que en su sinergia y combinación logran la funcionalidad de mostrar en el monitor cada carácter recibido desde el teclado. Es oportuno aclarar que este bloque por estar formado por 5 cajas con funcionalidades entrelazadas y básicas no se creyó necesario entra en detalles de diseño como en apartados anteriores. Esto no implica que sea el bloque funcional menos importante ni el más sencillo, es evidente viendo el anexo 16.3 más adelante que es el código más extenso del trabajo.

Los dos primeros bloques funcionales son simplemente convertidores de formato, es decir, convierten la trama de 10 bits recibida del teclado en formato *std_logic_vector* a un número *entero* para poder trabajar con el dato mucho más fácil y lograr la localización de cada pixel en pantalla más intuitivamente. Se diseñó uno para cada eje del monitor, uno para la *X* y otro para la *Y* [ver Figura 47].

La caja número 3 es un proceso cuya función es la de dividir la pantalla en las 32 columnas o sectores de 20x20 píxeles. Este selector toma el valor de *X* que proviene del bloque 1 y cada 20 píxeles la salida o señal *c* aumenta en '1' (son las columnas de la *1* a la *32*). La caja 4 es algo parecida a la 3, pero divide la pantalla en 24 filas o sectores de 20x20 píxeles también. De la misma forma que el bloque anterior coge el valor de la señal *Y* que viene del bloque 2 y lo junta en grupos de 20 filas, o sea, cada 20 filas la salida o señal *f* aumenta en '1' (son las filas de la *1* a la *24*). Combinando ambas cajas se logra la división virtual planteada o matriz de monitor, simplificando la acción de visualización o dibujado de los distintos caracteres [véase Figura 47].

Como último bloque se diseñó un proceso o especie de selector multicaso (parecido a una

[5] En este diagrama de bloques se han numerado las cajas para facilitar la explicación y detalle a posteriori, aunque se sabe que deberían tener sus propios nombres.

base de datos), con el objetivo de elegir o seleccionar cada patrón o sectores que hay que dibujar en la pantalla por cada trama de 8 bits (*Scancode*) recibida del teclado correspondiente a cada carácter identificado [ver Figura 47]. Estos patrones o tipografía se pueden ver detalladamente en la Tabla 19.

El Controlador o dibujador es un bloque que necesita obtener datos del Teclado *PS/2* y enviar otros al Monitor *VGA* para que funcione; esto implica que no se puede sintetizar, ni compilar en el *ISE Project Navigator* como un bloque independiente y así poder hacer el mismo análisis hecho con los bloques 1 y 2. Por ese mismo motivo sería muy complicado lograr una simulación de su implementación por separado y así poder ver que su funcionamiento es el correcto según lo esperado.

La comprobación del funcionamiento de este bloque y su análisis de la síntesis a fondo se estudia en el siguiente capítulo 12 de este proyecto, ya que engloba los tres diseños hechos hasta el momento. Concretamente el correcto funcionamiento en el apartado 12.2.5 y el análisis de la síntesis en el 12.2.4.

12. Bloque 4: General y comunicador de los bloques 1, 2 y 3.

12.1. Descripción del problema.

El cuarto y último bloque nombrado *Top Level* o General y comunicador, debe su nombre del anglicismo (nivel máximo o general). Este es el nivel superior en la jerarquía del diseño; en este bloque se materializan todos los conceptos e ideas tomadas en cuenta desde el comienzo de este proyecto. En este punto se consigue el funcionamiento global y la aplicación perseguida. La función de este *Top Level* es comunicar y transferir la información de un bloque a otro; es como el sustento o el canal de comunicación de los tres anteriores bloques, ya funcionales y probados (1, 2 y 3). La información se transmite desde el teclado por el puerto *PS/2* llega a la placa *Spartan 3E-Starter* se interpreta y analiza; y una vez codificada se envía a la pantalla *VGA* los píxeles exactos que hay que dibujar para presentar la información recibida desde teclado.

En el *Top Level* se utilizan todas las características con las que se ha trabajado hasta el momento de la *FPGA*, no se añade ninguna nueva. Sólo se pretende comprobar el buen comportamiento en conjunción de todos los diseños hechos. En caso necesario mejorar algún detalle para optimizar el rendimiento o aprovechamiento de la placa y analizar todos los resultados y comportamientos.

12.2. Especificaciones del bloque 4.

12.2.1. Interfaz.

En este último apartado se detallan cuáles son las entradas y salidas (E/S) de este bloque 4. Estas se pueden apreciar en el anexo del actual documento, exactamente en el anexo 16.4 más adelante. Las señales de este bloque son:

Entradas

- **ps2_clk**: es la señal del reloj que produce el teclado, la cual llega a la *FPGA* por el puerto *PS/2* y se modifica para adaptarse al reloj del *FPGA*. La frecuencia debe estar en el rango de 10 a 16.7 kHz como se especifica en el apartado 9.3.3.
- **ps2_dato**: es la señal que lleva a la *FPGA* los datos de cada tecla presionada o liberada a través del puerto *PS/2*. La actividad de esta señal está documentada en el apartado 9.3.
- **clk**: es la señal que representa el reloj de la *FPGA*, el cual fija todo el diseño. Tiene una frecuencia de 50 MHz lo que produce un periodo de 20 ns.
- **rst**: señal de inicialización asíncrona del sistema, activa a nivel alto.

Salidas

- **HSync**: señal de sincronismo horizontal, la cual se activa a los 656 ciclos de reloj siguiendo el protocolo *VGA*, activa a nivel bajo.
- **VSync**: señal de sincronismo vertical, la cual se activa a los 416 771 ciclos de reloj siguiendo el protocolo *VGA*, activa a nivel bajo.
- **R, G, B:** señales de 1 bit, cuya función es seleccionar el color a mostrar en pantalla.

12.2.2. Funcionalidad.

Con este bloque se logra el objetivo global y la aplicación perseguida. La tarea de este General y comunicador es enlazar y transmitir la información de un bloque a otro; actúa como la base de comunicación de los tres bloques anteriores (1, 2 y 3). La información se recibe desde el teclado mediante el protocolo *PS/2*, llega a la placa *Spartan 3E-Starter*, se interpreta y analiza; y una vez codificada se envía a la pantalla *VGA* la información precisa de los píxeles concretos que hay que mostrar o colorear para presentar la información recibida desde teclado.

Como se ha expuesto en el apartado anterior, este diseño por ser el más jerárquico contempla todo lo desarrollado. Esto implica que se usarán todas o la gran mayoría de las características de la placa *Spartan 3E-Starter Board* [observar Figura 48]. Todas no se usan debido a que al usar la técnica de diseño *bottom-up*, se van utilizando unas específicas para la creación y comprobación de cada bloque; que en el diseño general no tienen utilidad y tampoco sirven para comprobar el funcionamiento. Hecha esta aclaración, se puede observar en la Tabla 22 como se usan nueve elementos de la placa con sus respectivos pines, teniendo en cuenta que ya no se usa la señal *Error (A4)* como tampoco los ocho *LEDs (F12, E12, E11, F11, C11, D11, E9 y F9).* Específicamente estas características son utilizadas para la comprobación del funcionamiento del bloque *PS/2.*

De las nueve características la más importante y la que se usa en todo el diseño, es su *reloj principal*; cuyo valor de frecuencia es de 50 MHz, lo que implica que tenga un período de 20 ns. A nivel global también se implementa un *Reset* para volver al estado inicial por si hubiera algún problema o posible desajuste de funcionamiento. Además se trabaja con las señales de *reloj* y *dato* del puerto *PS/2*; sin dejar aparte las cinco señales que provienen del puerto *VGA*, las señales de *sincronismo vertical y horizontal* y las de los colores primarios *(R-G-B o Rojo-Verde-Azul)*. Todas las direcciones vienen en la Tabla 22 y sacadas del manual de la propia placa *Spartan*.

Elemento	Pin
Reloj	C9
Reset	L13
Reloj PS/2	G14
Dato PS/2	G13
HSync	F15
VSync	F14
Rojo	H14
Verde	H15
Azul	G15

Tabla 22. Pines de la *FPGA* para el bloque 4 (26).

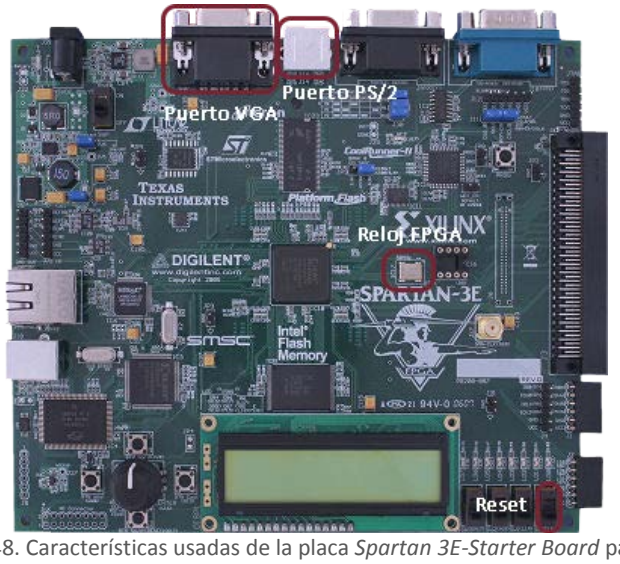


Figura 48. Características usadas de la placa *Spartan 3E-Starter Board* para el bloque 4 y último (26).

12.2.3. Arquitectura.

En este apartado sólo se muestran las partes o bloques del circuito, que se explicarán con más detalle en el siguiente apartado 12.2.4. En la Tabla 23 se enumeran todos los bloques del circuito y en la Figura 49 se puede ver el diagrama de bloques completo de la arquitectura.

Bloque	Descripción
1	Driver del teclado *PS/2*.
2	Driver de la pantalla *VGA*.
3	Controlador y dibujador.

Tabla 23. Resumen de los bloques del bloque 4.

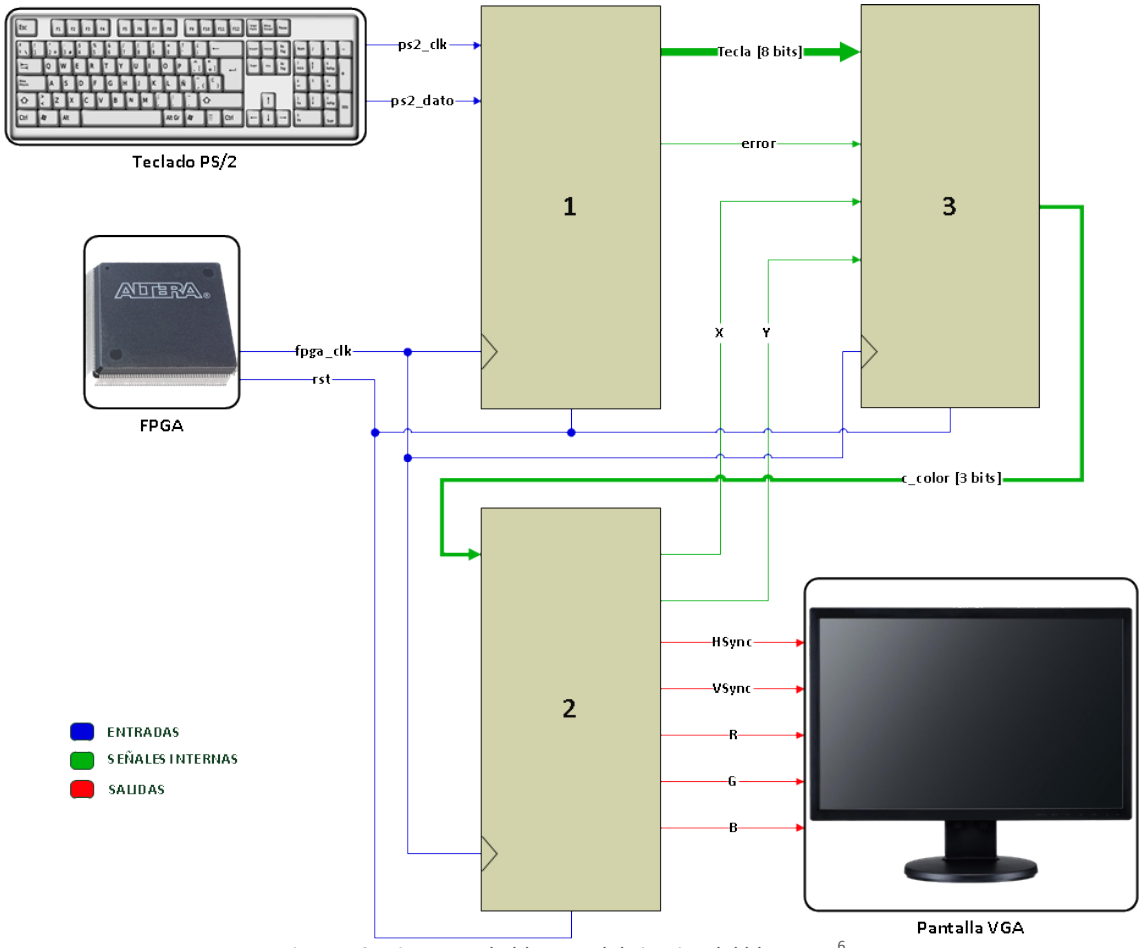


Figura 49. Diagrama de bloques del circuito del bloque 4.[6]

12.2.4. Diseño detallado y síntesis del circuito.

Este bloque final como se ha venido explicando con antelación y como se puede apreciar en la Tabla 23 es muy simple. Está formado por solo tres bloques de funcionamiento y estos son los que se han diseñado y explicado hasta el momento. El *Top Level* o General y comunicador es el nivel más alto en la jerarquía; éste sencillamente conecta o lleva las señales

[6] En este diagrama de bloques se han numerado las cajas para facilitar la explicación y detalle a posteriori, aunque se sabe que deberían tener sus propios nombres.

de un bloque a otro, es decir, le suministra a cada bloque en particular la información que requiere de los otros.

Llegado este punto se puede conseguir una idea más visual y funcional de todo el proyecto en su conjunto, viendo así más claro su objetivo global. Si se observa la Figura 49, se aprecia más esquemáticamente la distribución de las señales y la interconexión entre los bloques mencionados. Además de ver como entran en juego los periféricos usados y que señales aportan o se utilizan de los mismos. Del teclado mediante el puerto *PS/2* se obtienen dos señales *ps2_clk* y *ps2_dato*, que solamente son utilizadas por el bloque 1 que se encarga de interpretar y codificar la información que envía el teclado. De este primer bloque salen dos señales que van al bloque 3, una es la señal *Tecla* de 8 bits que es el código binario de la tecla presionada y la otra es *error*, que se utiliza para saber si hubo algún fallo de envío o de reconocimiento en el carácter.

Por otra parte al bloque 2 llega la señal *c_color* de 3 bits que proviene del bloque 3 y que es utilizada por el Driver de la pantalla *VGA* para saber qué color tiene que dibujar en cada momento. De este segundo bloque salen siete señales dos que van a la entrada del bloque 3 que son *X* e *Y* para posicionar y saber a cada instante que pixel se está procesando; y las otras cinco *HSync*, *VSync*, *R*, *G* y *B* que van directamente al monitor, cuyas funciones respectivamente son la de indicar cuando se ha completado una línea horizontal de la pantalla, cuando se ha recorrido una pantalla entera y la combinación de los 3 bits de los colores primarios para indicar el color a imprimir por pantalla. En este diseño sólo se utilizan tres colores blanco, negro y rojo.

Finalmente tenemos el tercer periférico que es la placa *Spartan 3E-Starter Board* de la cual los tres bloques toman las mismas señales, estas son la señal *fpga_clk* y *rst*. La primera es el reloj de la *FPGA* por la cual se rige todo el sistema y la segunda es el *reset* o reinicio asíncrono de dicho sistema, algo habitual y muy usado en cualquier diseño; ya sea como medida de reajuste o por fallos no deseados.

Como se especificó en el apartado 11.3.4 más atrás, es ahora que se han culminado los cuatro bloques del diseño que se puede sintetizar y probar el funcionamiento del bloque del Controlador o dibujador, así como también el comportamiento global. Al analizar todo el código con el *ISE Project Navigator* se generan algunos informes informativos que brindan información muy útil y habla de parámetros importantes del diseño, en este caso en particular del global.

Siguiendo la línea de análisis y verificación del proyecto llevado hasta el momento nos centramos en el resumen de síntesis, donde se puede apreciar el porcentaje de utilización de la *FPGA*, es decir, muestra el número de unidades lógicas utilizadas por el diseño total con respecto al total disponible en la *FPGA* (*Flip-Flops*, *LUTs* o *Slices*).

Observando la imagen siguiente [Figura 50] y entrando en detalles el análisis obtenido demuestra que el número de *Slices* ocupados es 524 de 4656 disponibles, lo que representa un 11% de utilización respecto al total disponible. La cantidad de *Flip-Flops* usados es 76 de 9312, equivalente al 1% de utilización del disponible y el número de *4 input LUTs* es 960 de 9312 que corresponde al 10% del total disponible. El número de *bonded IOBs* es 9 de 232 que es el 3%. El número de *BUFGMUXs* es 1 de 24 que es el 4% del total.

Device Utilization Summary			
Logic Utilization	**Used**	**Available**	**Utilization**
Number of Slice Flip Flops	76	9,312	1%
Number of 4 input LUTs	960	9,312	10%
Number of occupied Slices	524	4,656	11%
Number of Slices containing only related logic	524	524	100%
Number of Slices containing unrelated logic	0	524	0%
Total Number of 4 input LUTs	989	9,312	10%
Number used as logic	960		
Number used as a route-thru	29		
Number of bonded IOBs	9	232	3%
Number of BUFGMUXs	1	24	4%
Average Fanout of Non-Clock Nets	3.62		

Figura 50. Resumen de la utilización de la *FPGA* bloque 4.

Del Resumen de Tiempos, se obtienen otros datos que hay que tener muy en cuenta si quisiera utilizarse este diseño en otras placas o *FPGAs*. Los tiempos globales que se obtuvieron se muestran a continuación en la Tabla 24:

Minimum period: 14.602ns (Maximum Frequency: 68.483MHz)
Minimum input arrival time before clock: 7.838ns
Maximum output required time after clock: 8.787ns
Maximum combinational path delay: No path found

Tabla 24. Extracto del informe de tiempos de la *FPGA* bloque 4.

Analizando con detenimiento los datos anteriores tenemos que el período mínimo de una señal es de 14,602 ns, esto conlleva a que la frecuencia máxima sea de 68,483 MHz. Básicamente, esto demuestra que si se pretende utilizar el diseño actual en otra *FPGA* habría que garantizar que ésta tiene un reloj con una frecuencia igual o menor que la frecuencia calculada, de otra manera no funcionaría el circuito de la forma deseada. Por otro lado se muestra que el tiempo mínimo que demora en llegar una señal de entrada antes de un pulso de reloj es de 7,838 ns y además que el tiempo máximo que requiere una señal de salida después de un pulso de reloj es de 8,787 ns.

También tenemos que el retraso máximo en una ruta combinacional no se encuentra, que en este caso esto se interpreta que como estamos en el nivel más alto en la jerarquía del diseño y sólo se conectan las entradas y salidas de los bloques anteriores no se crean rutas combinacionales muy complejas que puedan provocar algún retraso. Es extremadamente importante cumplir con estas especificaciones como se ha dicho anteriormente si se desea emplear este diseño en otras *FPGAs*. Cumpliendo con los tiempos y las frecuencias especificadas anteriormente aseguramos un funcionamiento correcto y eficaz.

12.2.5. Descripción del banco de pruebas del bloque 3 y 4.

Como se mencionó en el apartado 11.3.4 más atrás, ahora que se han culminado los cuatro bloques del diseño es que se puede probar el funcionamiento del bloque del Controlador o dibujador. Es imprescindible dejar claro que si se han podido hacer las siguientes simulaciones es porque el bloque 4 funciona a la perfección.

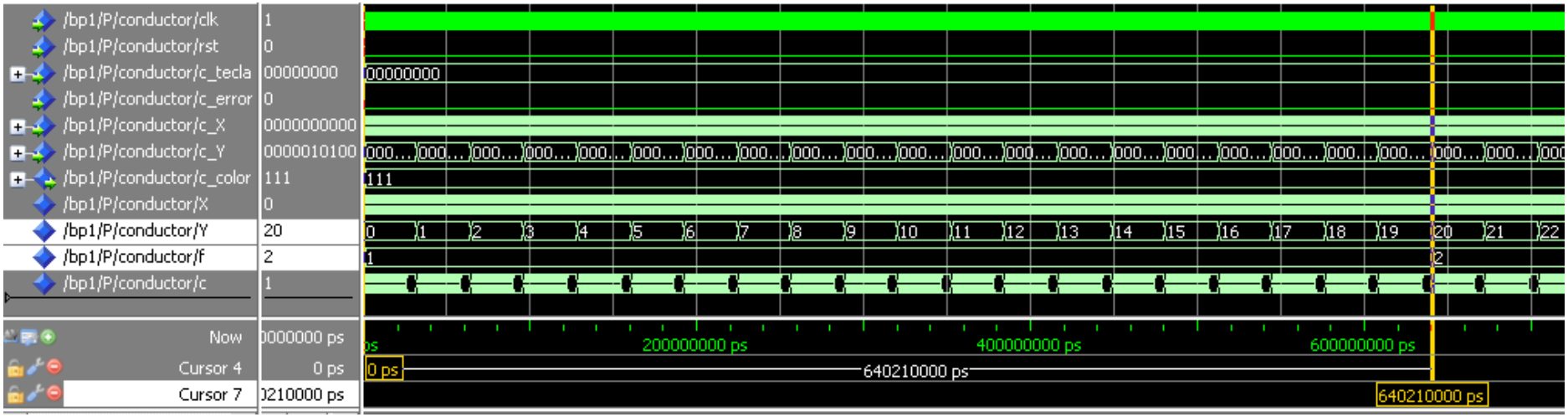


Figura 51. Simulación 22.

Esta simulación está enfocada a comprobar lo que se explica en el apartado 11.3.4 con respecto a la conformación de filas. Como se puede ver en la imagen cuando la señal *Y* toma el valor de 0 a 19 estamos hablando de la fila primera, es decir, cada 20 líneas horizontales se conforma una fila.

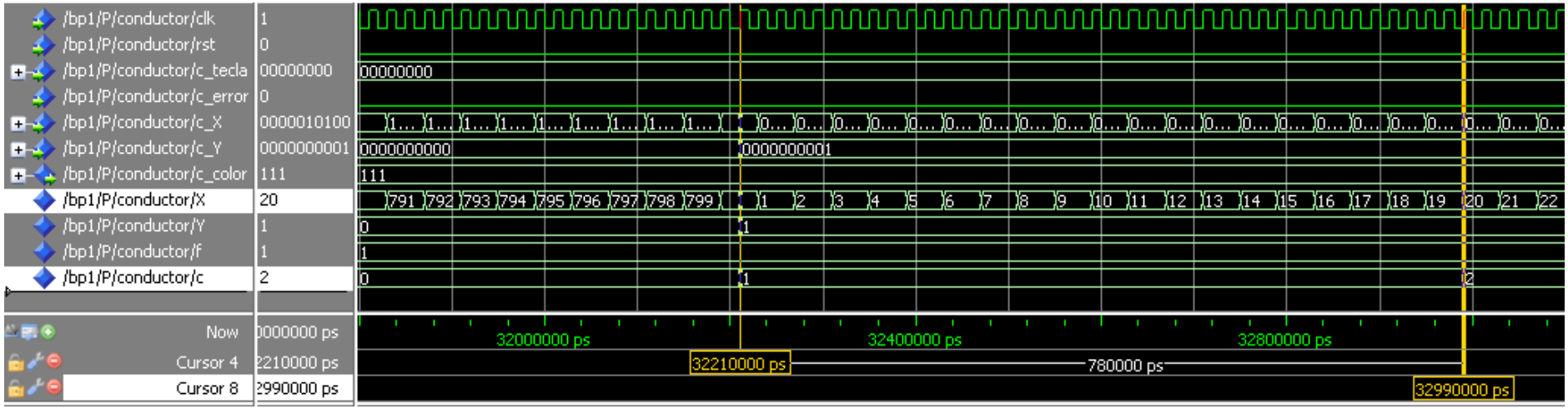


Figura 52. Simulación 23.

Esta simulación está enfocada a comprobar lo que se explica en el apartado 11.3.4 con respecto a la conformación de columnas. Como se puede ver en la imagen cuando la señal *X* toma el valor de 0 a 19 estamos hablando de la columna primera, es decir, cada 20 píxeles se conforma una columna.

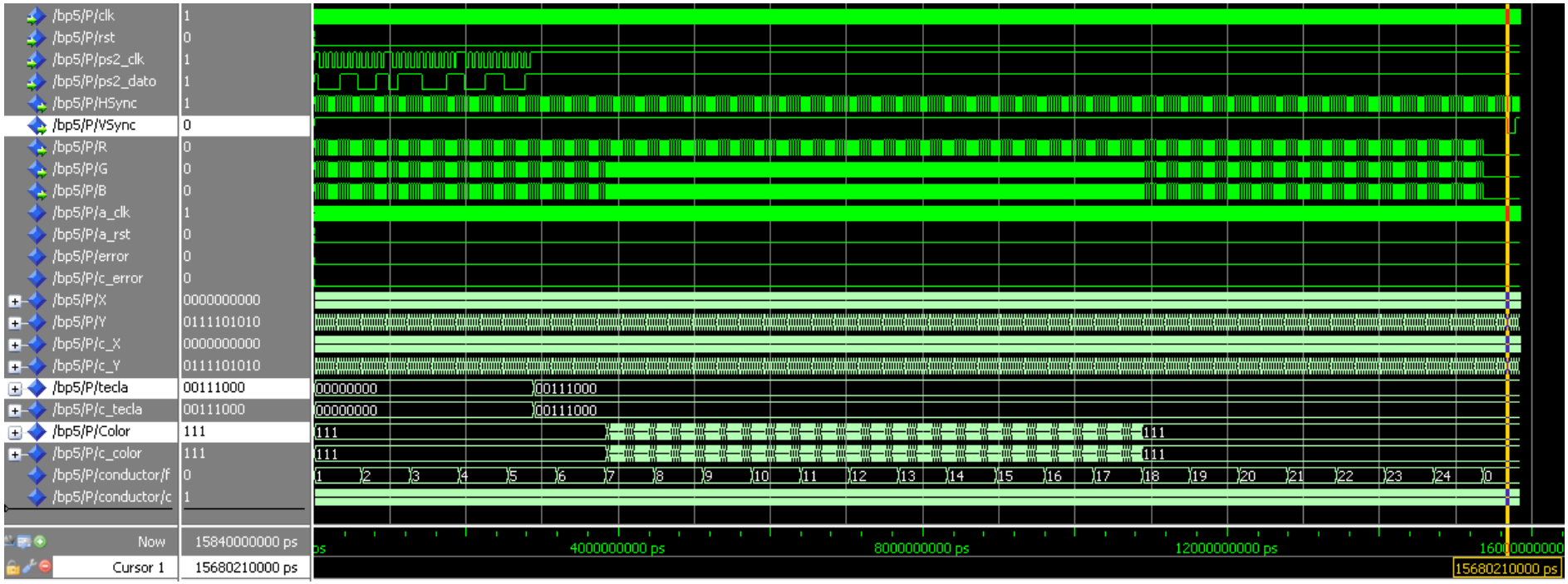


Figura 53. Simulación 24.

En esta simulación se muestra como es el comportamiento general del sistema cuando se presiona la tecla *'A'* = "1C h = 0001 1100 b". En primer lugar se aprecia que el *scancode* del carácter *'A'* se registra sin problemas y se guarda hasta que exista otro envía desde teclado. Por otra parte, vemos que a modo general la señal *Color* cambia su valor en dependencia de lo que le asigne el bloque Controlador o dibujador. Por último, queda muy claro que se completa la presentación en pantalla del carácter recibido porque la señal *VSync* se activa y esto sólo sucede si se termina de imprimir una pantalla entera.

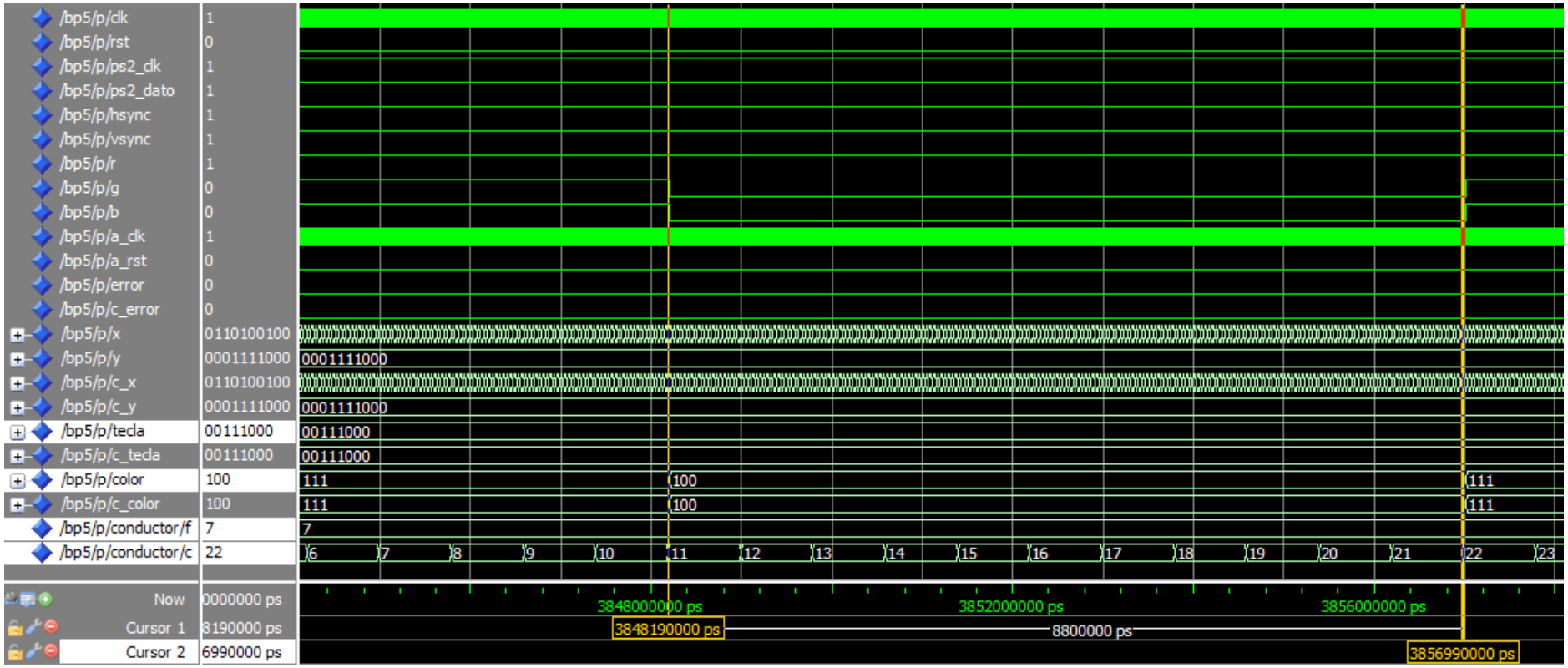


Figura 54. Simulación 25.

Esta simulación se hizo para comprobar el comportamiento del bloque Controlador o dibujador cuando se presiona la tecla *'A'* = "1C h = 0001 1100 b". Especialmente en esta imagen se muestra como se imprime en pantalla el marco rojo centrado y el fondo blanco, como se explica en el apartado 11.2 más atrás. Observando la Tabla 12, la Figura 45 y ésta simulación se sabe que los sectores de la fila 7 con las columnas de la 11 a la 21 se colorean de rojo; el resto de la fila 7 se dibuja con el color blanco y esto representa el marco rojo superior.

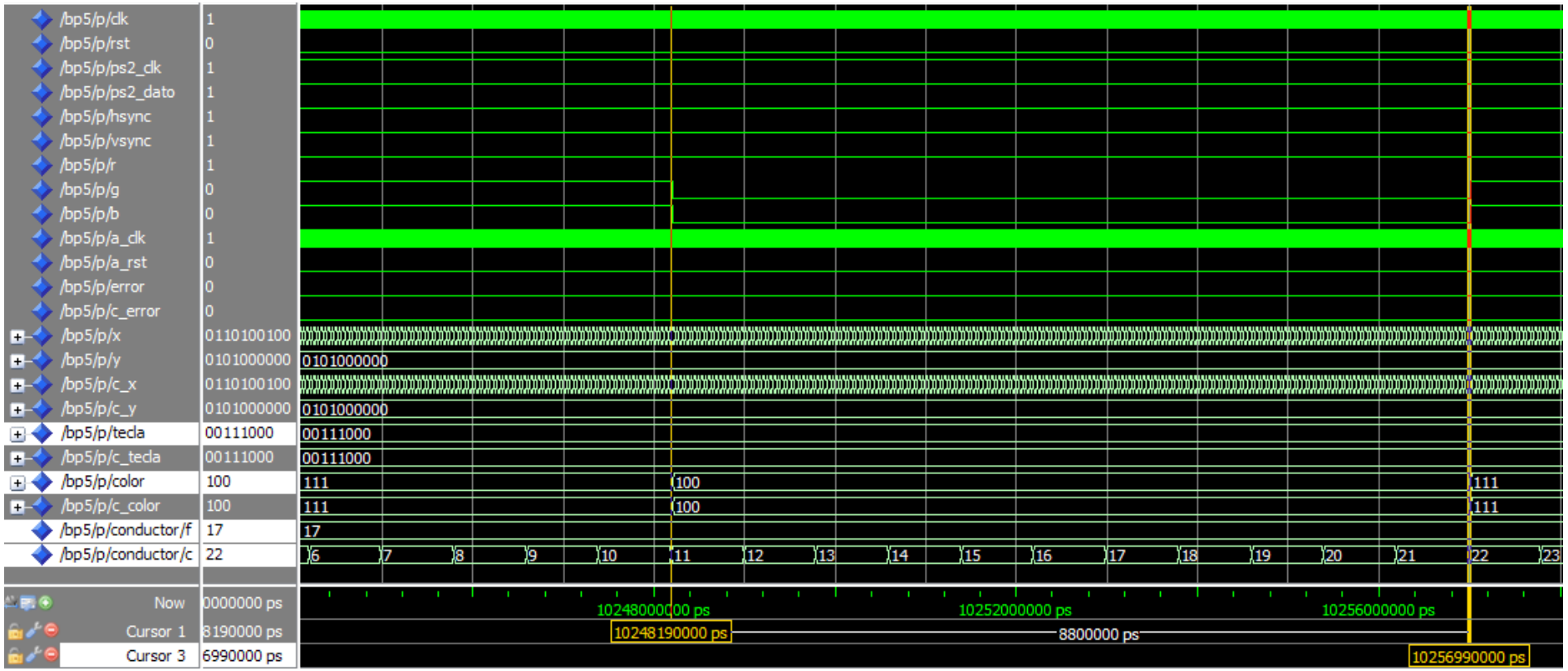


Figura 55. Simulación 26.

Esta simulación se hizo para comprobar el comportamiento del bloque Controlador o dibujador cuando se presiona la tecla *'A'* = "1C h = 0001 1100 b". Especialmente en esta imagen se muestra como se imprime en pantalla el marco rojo centrado y el fondo blanco, como se explica en el apartado 11.2 más atrás. Observando la Tabla 12, la Figura 45 y ésta simulación se sabe que los sectores de la fila 17 con las columnas de la 11 a la 21 se colorean de rojo; el resto de la fila 17 se dibuja con el color blanco y esto representa el marco rojo inferior.

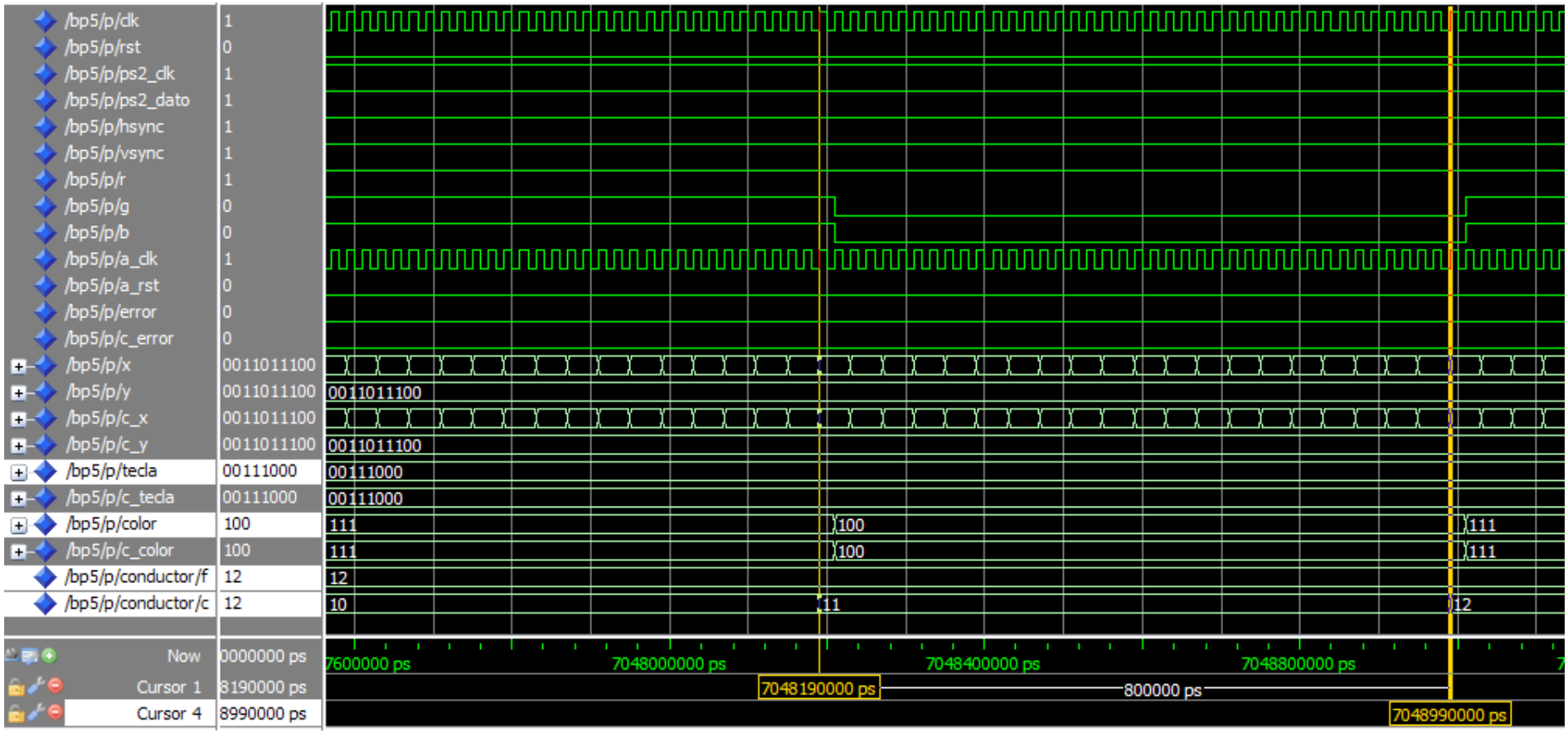


Figura 56. Simulación 27.

Esta simulación se hizo para comprobar el comportamiento del bloque Controlador o dibujador cuando se presiona la tecla *'A'* = "1C h = 0001 1100 b". Especialmente en esta imagen se muestra como se imprime en pantalla el marco rojo centrado y el fondo blanco, como se explica en el apartado 11.2 más atrás. Observando la Tabla 12, la Figura 45 y ésta simulación se sabe que el sector de la fila 12 con la columna 11 al igual que el de la fila 12 con la columna 21 se colorean de rojo; el resto de la fila 12 se dibuja con el color blanco y esto se repite exactamente igual para las filas de la 8 a la 16, representando los marcos laterales rojo.

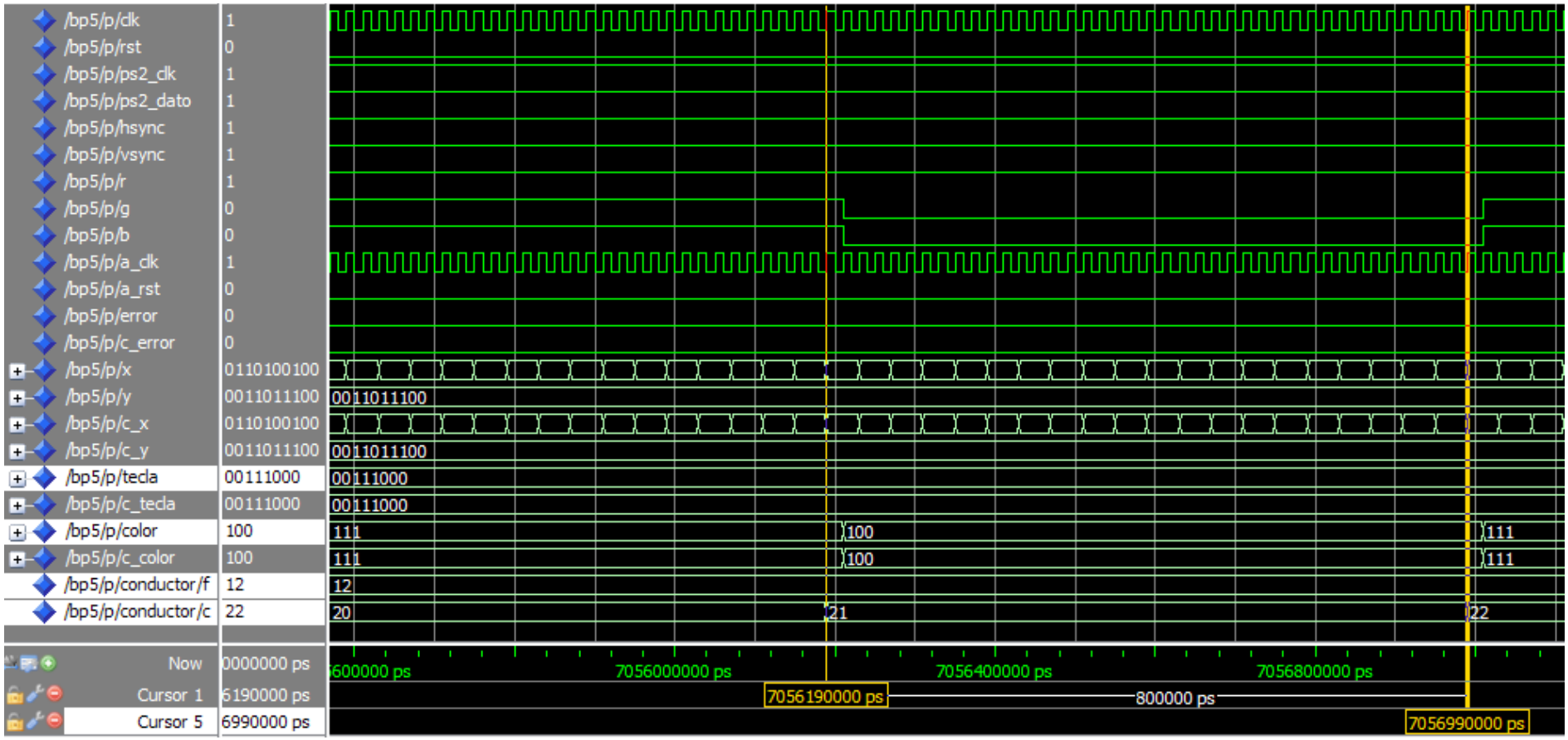


Figura 57. Simulación 28.

Esta simulación se hizo para comprobar el comportamiento del bloque Controlador o dibujador cuando se presiona la tecla *'A'* = "1C h = 0001 1100 b". Especialmente en esta imagen se muestra como se imprime en pantalla el marco rojo centrado y el fondo blanco, como se explica en el apartado 11.2 más atrás. Observando la Tabla 12, la Figura 45 y ésta simulación se sabe que el sector de la fila 12 con la columna 11 al igual que el de la fila 12 con la columna 21 se colorean de rojo; el resto de la fila 12 se dibuja con el color blanco y esto se repite exactamente igual para las filas de la 8 a la 16, representando los marcos laterales rojo.

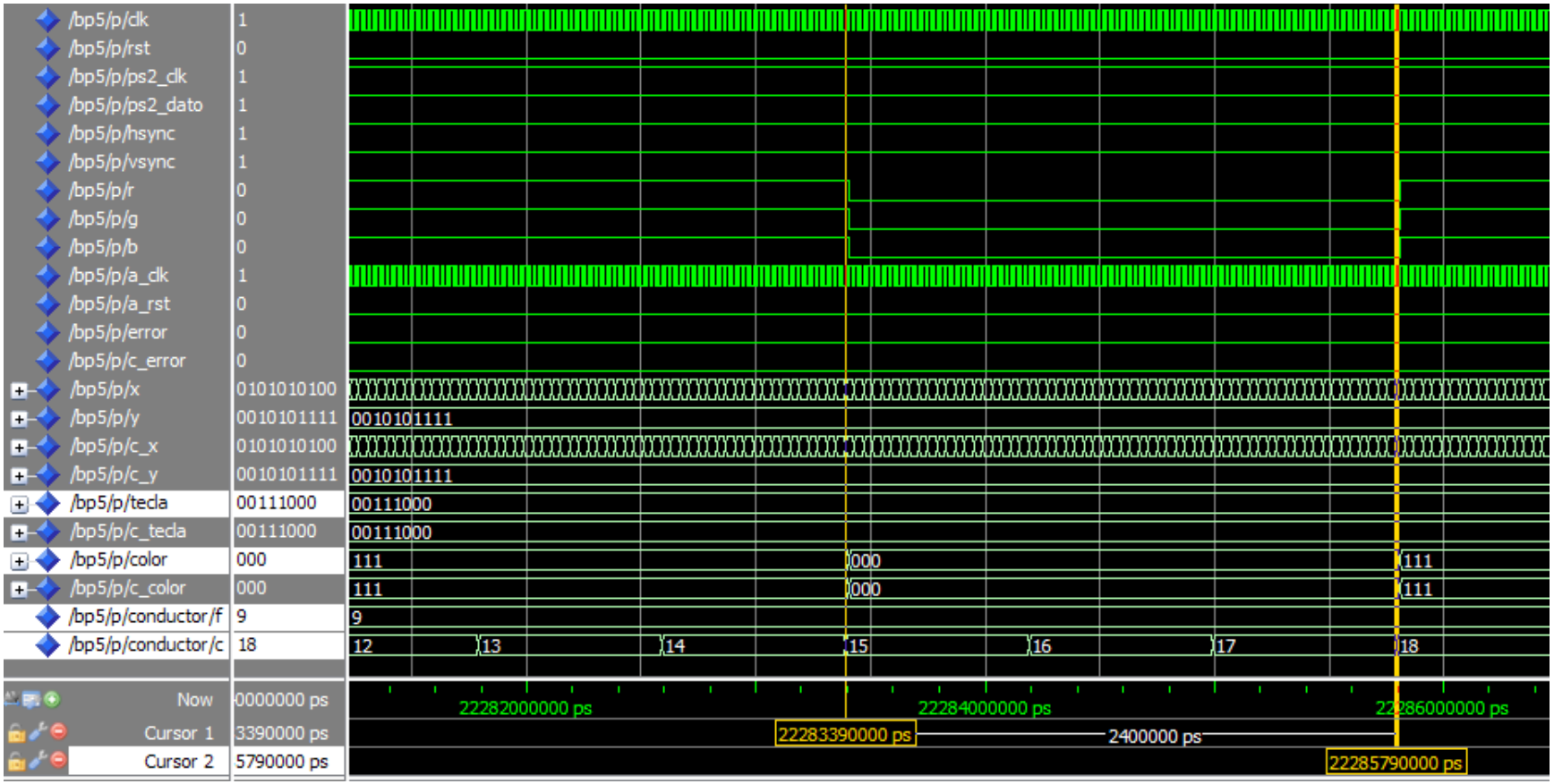


Figura 58. Simulación 29.

Esta simulación se hizo para comprobar el comportamiento del bloque Controlador o dibujador cuando se presiona la tecla *'A'* = "1C h = 0001 1100 b". Especialmente en esta imagen se muestra como se imprime en pantalla la tipografía definida en color negro, como se explica en el apartado 11.2 más atrás. Observando la Tabla 12, la Tabla 19 y ésta simulación se sabe que los sectores de la fila 9 con las columnas de la 15 a la 17 se colorean de negro; el resto de la fila 9 se dibuja con el color blanco y esto representa la línea horizontal superior negra del carácter *'A'*.

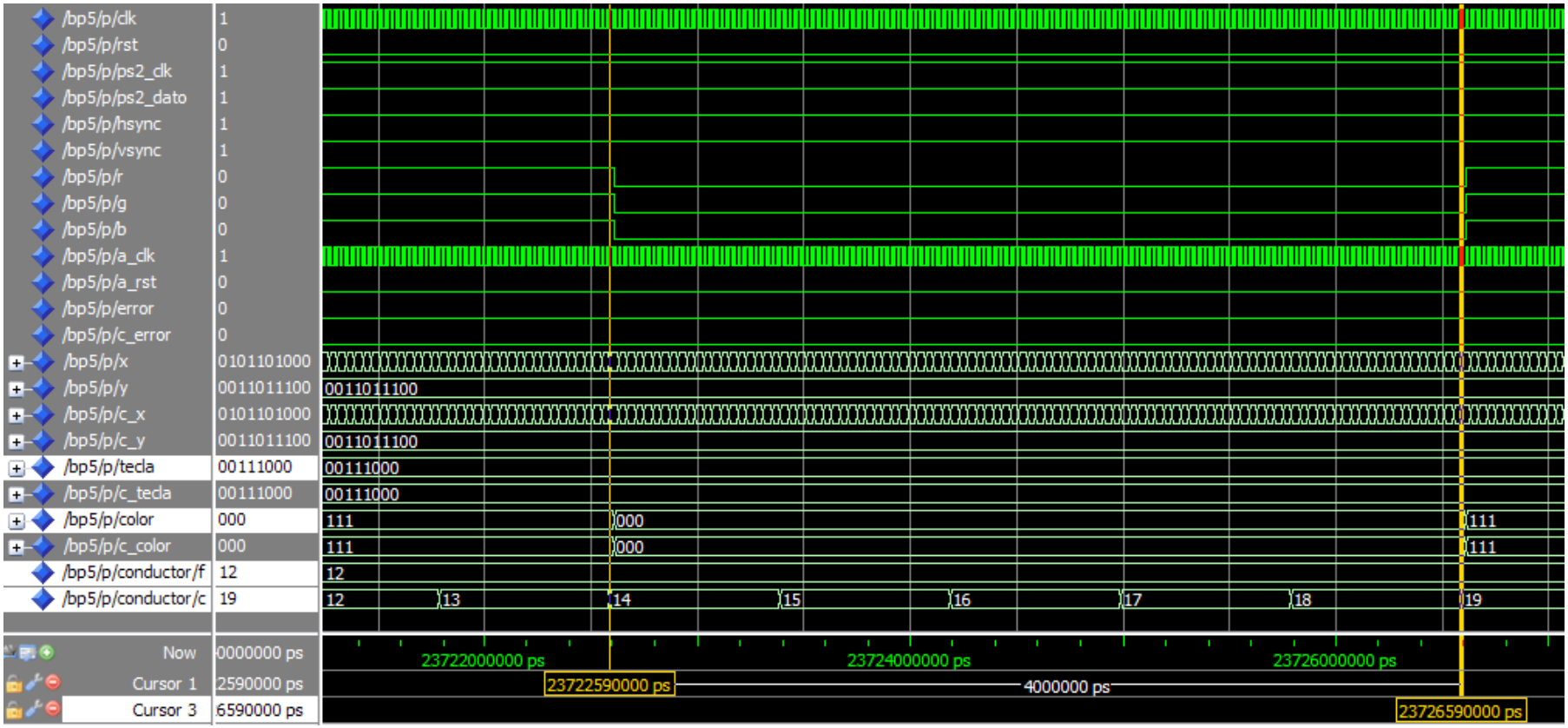


Figura 59. Simulación 30.

Esta simulación se hizo para comprobar el comportamiento del bloque Controlador o dibujador cuando se presiona la tecla *'A'* = "1C h = 0001 1100 b". Especialmente en esta imagen se muestra como se imprime en pantalla la tipografía definida en color negro, como se explica en el apartado 11.2 más atrás. Observando la Tabla 12, la Tabla 19 y ésta simulación se sabe que los sectores de la fila 12 con las columnas de la 14 a la 18 se colorean de negro; el resto de la fila 12 se dibuja con el color blanco y esto representa la línea horizontal del centro negra del carácter *'A'*.

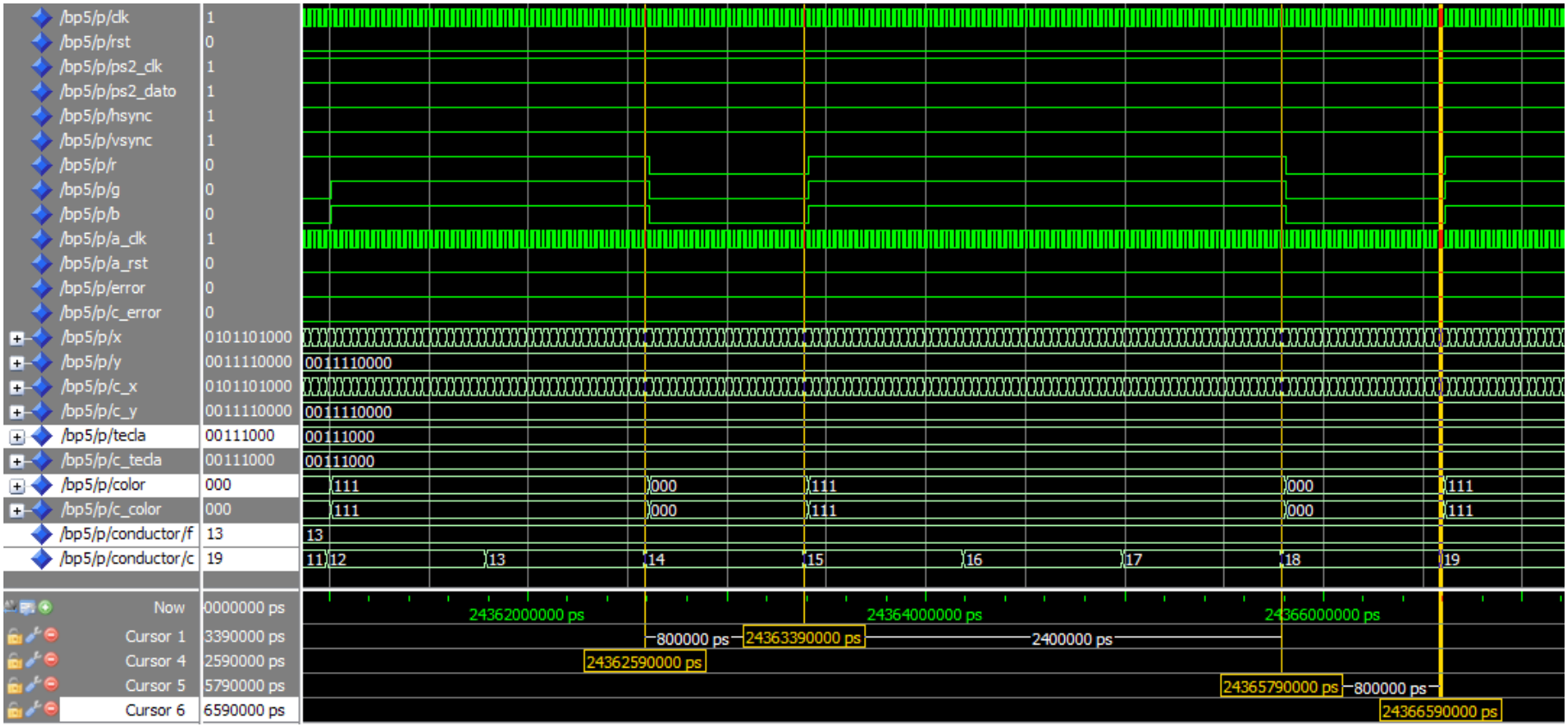


Figura 60. Simulación 31.

Esta simulación se hizo para comprobar el comportamiento del bloque Controlador o dibujador cuando se presiona la tecla *'A'* = "1C h = 0001 1100 b". Especialmente en esta imagen se muestra como se imprime en pantalla la tipografía definida en color negro, como se explica en el apartado 11.2 más atrás. Observando la Tabla 12, la Tabla 19 y ésta simulación se sabe que el sector de la fila 13 con la columna 14 al igual que el sector de la fila 13 con la columna 18 se colorean de negro; el resto de la fila 13 se dibuja con el color blanco y esto se repite exactamente igual para las filas 10, 11, 14 y 15. Representando las líneas verticales laterales negras del carácter *'A'*.

13. Conclusiones y trabajos futuros.

De la realización del presente proyecto se obtienen una serie de resultados relevantes y además varias ideas para darle continuidad a este trabajo en un futuro y completar un funcionamiento convencional.

13.1. Conclusiones.

Primeramente y volviendo a los objetivos planteados al comienzo del proyecto, hablamos de 4 específicos con los que se conseguían la funcionalidad global. El primer objetivo específico era diseñar e implementar la interfaz y protocolo de comunicación del teclado *PS/2* estándar, Dicho objetivo se alcanzó satisfactoriamente y se podría decir que con un funcionamiento óptimo, ya que durante todas las pruebas y comprobaciones hechas no se obtuvo nunca fallo alguno a la hora de capturar el código de escaneo de ninguna tecla. Esto se explica porque a pesar de los rebotes reales que existen al presionar una tecla y la repetición del mismo código mientras esta pulsada, el diseño sólo toma y registra el último código recibido y es el con el que se trabaja.

El siguiente objetivo específico logrado dentro del proyecto, fue diseñar e implementar la interfaz y protocolo de comunicación de la pantalla o monitor *VGA*. Respecto a este objetivo, se puede decir que se cumplen con todos los tiempos de sincronización de las señales necesarias del protocolo. Además que el diseño creado se podría usar para diferentes resoluciones, simplemente cambiando los valores de las señales de sincronización.

A la hora de diseñar e implementar el controlador y dibujador [DRIVER] para controlar la tecla recibida del teclado *PS/2* y sacarla por pantalla *VGA* constantemente hasta que se presione la siguiente tecla o en su caso exista un *Reset* manual; fue una tarea que al principio no resultó ser trivial, ni tampoco fácil de captar cual sería el mejor funcionamiento. Esto provocó cierto tiempo de asimilación del algoritmo a seguir, pero una vez interiorizado y esquematizado dicho algoritmo, su desarrollo posterior no fue algo excesivamente complicado pero si muy trabajoso y en ocasiones algo denso de lograr. Básicamente esto se debe a que por cada carácter a mostrar en el monitor hay que definir toda una pantalla cubriendo la totalidad de los pixeles con sus respectivos colores. Lo que se traduce en numerosas pruebas y ensayos-error para lograr el resultado buscado para cada carácter o tecla. Este código sin duda es el más extenso con diferencia respecto a los demás, sin embargo es el más sencillo de entender y analizar debido a la definición previa de la interfaz de la pantalla *VGA*.

Lograr la comunicación entre todos los diseños fue un hecho que prácticamente se produjo sólo debido a la estupenda conceptualización y cumplimiento de los protocolos exigidos. Fue una tarea que apenas significó tiempo de elaboración, pero si mucha labor de comprobación y verificación. Este es el cuarto y último objetivo específico conseguido, además de ser el objetivo fundamental del presente proyecto fin de carrera y al cual da título. De forma general y concluyente se puede decir que se cumplió con la funcionalidad propuesta en el proyecto que era desarrollar una interfaz de comunicación entre un teclado *PS/2* convencional y una pantalla o monitor *VGA*. Todo esto es debido al cumplimiento por separado de todos los objetivos parciales de una forma robusta y funcional.

El logro más significativo de este proyecto es el desarrollo de la interfaz de teclado *PS/2*, esto era algo que se buscaba en el departamento de Tecnología Electrónica de la Universidad Carlos III de Madrid y que hasta el momento no se había desarrollado, ni implementado. Otro logro significativo de este proyecto es que todo el diseño es síncrono e independiente de la tecnología, esto quiere decir, que se podría implementar en cualquier otra FPGA y funcionaría a la perfección siempre que se mantenga la misma frecuencia de reloj. Como se ha dicho con anterioridad esto tendrá mucha aplicación a nivel educativo, además de ser utilizado por el

propio departamento para desarrollar futuros proyectos usando estos protocolos e interfaces.

13.2. Mejoras y trabajos futuros.

Dentro de todo proyecto siempre se desea que haya una mejora continua del mismo; por lo tanto se recomienda a futuros proyectantes que tengan interés en el proyecto, la complementación y ampliación del sistema diseñado hasta el momento con la incorporación de las segundas funciones de teclas e incluso las teclas especiales o multimedias de teclados más modernos. También se puede completar el diseño con la implementación de la comunicación bidireccional desde el *host* al teclado y conseguir los estados de *"Request to Send"* y *"Comunicación Inhibida"*. Otra recomendación podría ser, buscar un método o algoritmo para mostrar los caracteres en pantalla menos engorroso o denso, e incluso desarrollar el diseño para poder alcanzar una escritura continua como en un editor de texto estándar.

14. Lista de Acrónimos.

- FPGA (*Field Programmable Gate Array* o Campo de Matriz de Puertas Programables).
- VGA (*Video Graphics Array* o Matriz Gráfica de Video).
- PS/2 (*Personal System/2* o Sistema Personal/2).
- VHDL (*Very High Speed Integrated Circuit* o Circuitos Integrados de Muy Alta Velocidad).
- LED (*Light Emitting Diode* o Diodo Emisor de Luz).
- RAM (*Random Access Memory* o Memoria de Acceso Aleatorio).
- SRAM (*Static Random Access Memory* o Memoria Estática de Acceso Aleatorio).
- DRAM (*Dynamic Random-Access Memory* o Memoria de Acceso Aleatorio Dinámica).
- NAND (*Negated AND* o AND Negada).
- PAL (*Programmable Array Logic* o Lógica de Matriz Programable).
- CPLD (*Complex Programmable Logic Device* o Dispositivo Lógico Programable Complejo).
- DSP (*Digital Signal Processing* o Procesamiento Digital de Señales).
- ASIC (*Application Specific Integrated Circuit* o Circuito Integrado de Aplicación Específica).
- HDL (*Hardware Description Language* o Lenguaje de Descripción Hardware).
- ABEL (*Advanced Boolean Expression Language* o Lenguaje Avanzado de Expresión Booleano).
- VHSIC (Very High Speed Integrated Circuit o Circuitos Integrados de Muy Alta Velocidad).
- IEEE (*Institute of Electrical and Electronics Engineers* o Instituto de Ingenieros Eléctricos y Electrónicos).
- ANSI (*American National Standards Institute* o Instituto Nacional de Estándares Americano).
- IBM (*International Business Machines* o Máquinas Internacionales de Negocios).
- PC (*Personal Computer* u Ordenador Personal).
- USB (*Universal Serial Bus* o Bus Universal en Serie).
- PC/XT (*Personal Computer with Extended Technology* u Ordenador Personal con Tecnología Extendida).
- PC/AT (*Personal Computer with Advanced Technology* u Ordenador Personal con Tecnología Avanzada)
- TFT (*Thin Film Transistor* o Transistor de Película Fina).
- LCD (*Liquid Crystal Display* o Pantalla de Cristal Líquido).
- EGA (*Enhanced Graphics Adapter* o Adaptador Gráfico Mejorado).
- DAC (*Digital to Analog Converter* o Convertidor Digital - Analógico)
- BIOS (*Basic Input/Output System* o Sistema de Entrada/Salida Básico)
- CRT (*Cathode Rays Tube* o Tubo de Rayos Catódicos).
- UV (*Ultra Violet* o Ultra Violeta).
- IP Cores o IPC (*Intellectual Property Cores* o Núcleos de Propiedad Intelectual).
- ATPG (*Automatic Test Pattern Generation and Automatic Test Pattern Generator* o Modelo de Prueba de Generación Automática y Generador Automático de Patrones de Test).
- ASCII (*American Standard Code for Information Interchange* o Código Americano Estándar para el Intercambio de Información).
- LUTs (*Look-up Tables* o Tablas de Consulta).
- RISC (*Reduced Instruction Set Computer* o Computador con Conjunto de Instrucciones

Reducidas)
- CPU (*Central Processing Unit* o Unidad Central de Procesamiento)

15. Glosario de Términos

- Host: es una palabra inglesa cuyo significado es "anfitrión(ona), huésped o presentador(a)". En este proyecto se utiliza esa palabra para designar al dispositivo central, que es la *FPGA*.
- Scancode: es toda la información que el teclado envía a la *FPGA* ya sea de cualquier tecla que se presiona, se libera o se mantiene pulsada. Existen dos tipos diferentes de *scancodes*: *makecode* y *breakcode*.
- Makecode: es el paquete de información que se envía a la *FPGA* cuando se pulsa una tecla o se mantiene presionada.
- Breakcode: es el paquete de información que se envía a la *FPGA* cuando se suelta o se libera una tecla presionada.
- Typematic: cuando se presiona y mantiene presionada una tecla, esa tecla se convierte en *typematic*, lo que significa que el teclado seguirá enviando el *makecode* de la misma hasta que la tecla se libera o se pulsa otra.
- PowerPC: (abreviatura usada *PPC*) son procesadores cuyo nombre original proviene de la arquitectura computacional de tipo *RISC*, que fue desarrollada por *IBM*, *Motorola* y *Apple*. Esta arquitectura *RISC* es un tipo de diseño de *CPU* generalmente utilizado en microprocesadores o microcontroladores con ciertas características de propósito general.
- Microcontroladores: es un circuito integrado programable, capaz de ejecutar las órdenes grabadas en su memoria.
- Motherboard: es conocida comúnmente como la tarjeta madre o placa base. Es una tarjeta de circuito impreso que permite la integración de todos los componentes de un ordenador. Para esto, cuenta con un software básico conocido como *BIOS*, que le permite cumplir con sus funciones.
- Keyboard: en español es el teclado y es un dispositivo de entrada de datos.
- Mouse: en español es el ratón y es un dispositivo de entrada de datos.
- Windows: sistema operativo desarrollado por la compañía *Microsoft*.
- MacOS: sistema operativo desarrollado por la compañía *Apple*.
- Linux: sistema operativo de software libre y compatible con *UNIX*.
- Joystick: es una palanca de mando y es un dispositivo de control de dos o tres ejes que se usa desde un ordenador o videoconsola hasta un transbordador espacial (incluso aviones).
- Flag: termino que proviene del inglés y que se utiliza en el mundo de la programación para referirse a una especie de marcador en un punto específico del código.
- Plug and Play: Término que se usa comúnmente para nombrar al tipo de interfaz de fácil conexión y desconexión.
- Verilog: lenguaje de descripción de hardware usado para modelar sistemas electrónicos.
- S3 Graphics, NVIDIA o ATI: fabricantes de tarjetas gráficas o de video para ordenadores.
- Key Pad: forma de nombrar al teclado numérico de un teclado convencional en inglés.
- Flip-Flop: es un biestable o un multivibrador capaz de permanecer en uno de dos estados posibles durante un tiempo indefinido en ausencia de perturbaciones.
- Slices: palabra del idioma inglés cuyo significado es parte, porción, rebanada, rodaja, tajada o loncha.

16. Anexos.

En esta sección simplemente se proporciona el código *VHDL* de los circuitos diseñados.

16.1. Código *VHDL* del Teclado *PS/2*.

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_ARITH.ALL;
entity teclado107 is
  port(ps2_clk   : in  std_logic;--reloj del puerto PS/2.
       ps2_dato  : in  std_logic;--datos del puerto PS/2.
       clk       : in  std_logic;--reloj principal de la fpga (50MHz)->T=20ns.
       rst       : in  std_logic;--reinicio global del sistema.
       error     : out std_logic;--señal de salida de error.
       tecla     : out std_logic_vector(7 downto 0));--codigo hexadecimal de la tecla
                                                     --presionada.
end teclado107;
architecture BEH of teclado107 is
-------------Declaracion de señales-----------------
signal ps2_clk_in        : std_logic;--señal del reloj que envia el dispositivo(teclado)
                                     --al host(FPGA).
signal DB                : std_logic_vector(1 downto 0);--señal de 2 bits de los ff del
                                                        --detector de flancos de bajada.
signal DS                : std_logic_vector(1 downto 0);--señal de 2 bits de los ff del
                                                        --detector de flancos de subida.
signal fb                : std_logic;--señal que detecta los flancos de bajada del reloj
                                     --del PS/2.
signal fs                : std_logic;--señal que detecta los flancos de subida del reloj
                                     --del PS/2.
signal flanco            : std_logic;--señal para saber cuando hay un flanco, ya sea de
                                     --subida o bajada.
signal ps2_clk_HL        : std_logic;--señal que se activa cada 50us y un ciclo de reloj
                                     --mas del reloj del host(FPG), despues de un flanco
                                     --de subida.
signal cuentaHL_50us     : integer range 0 to 2500;--señal que cuenta los ciclos
                                                   --suficientes para llegar hasta
                                                   --50us=2499 y un ciclo mas en el
                                                   --TEMP_50usHL_ps2_clk.
signal fcuentaHL_50us    : std_logic;--señal que sirve para reiniciar el temporizador
                                     --cuando llega a su limite.
signal error_a,error_r   : std_logic;--señales auxiliares que se activan cuando hay
                                     --algun error en la comunicacion por cualquier
                                     --causa.
signal makecode          : std_logic;--señal que se activa cuando se recibieron 11 bits
                                    --y se detiene el reloj del PS/2 a nivel
                                    --alto (scancode).
signal breakcode         : std_logic;--señal que se activa cuando se recibio un
                                     --breakcode(se libera la tecla).
signal cuenta_11fb       : integer range 0 to 11;--señal que cuenta los 11 ciclos de
                                                 --cada dato enviado.
signal fcuenta_11fb      : std_logic;--señal que sirve para reiniciar el contador cuando
                                     --llega a su limite.
signal RD                : std_logic_vector(0 to 9);--señal de 10 bits de los ff del
                                                    --registro de desplazamiento,son
                                                    --solo 10 bits porque el ultimo
                                                    --bit de final no lo utilizo.
signal raux,salida       : std_logic_vector(0 to 7);--señales de 8 bits de los ff de los
                                                    --registros auxiliar y de salida
                                                    --para almacenar el scancodes.
signal guarda,save       : std_logic;--señal que sirve como enable de los registros de 8
                                     --bits para guardar los scancodes.
type FSM is (T_pres,T_libe,Imprime_tecla);--estados de la máquina de estados.
Signal FSM_EA,FSM_ES : FSM;--señales que indican el estado actual y el siguiente de la
                           --FSM.
Begin
-------------------------------------------------------------------------------------
-------------------------------------------------------------------------------------
ps2_clk_in <= ps2_clk;--asigno el reloj del PS/2 a la señal de reloj de entrada al host.
-------------------------------------------------------------------------------------
flanco <= fs or fb;--detecta cualquier flanco,de subida o bajada.
-------------------------------------------------------------------------------------
```

```
ps2_clk_HL <= '1' when (cuentaHL_50us = 2500 and ps2_clk_in = '1') else '0';--señal que
                                                                            --se activa
                                                                            --cuando se
                                                                            --detiene
                                                                            --el reloj
                                                                            --del PS/2 y
                                                                            --esta a
                                                                            --nivelalto.
-------------------------------------------------------------------------------------------
makecode <= '1' when (cuenta_11fb = 11 and ps2_clk_HL = '1') else '0';--detector de
                                                                      --scancode
                                                                      --recibido.
-------------------------------------------------------------------------------------------
breakcode <= '1' when raux = "11110000" else '0';--se activa cuando el registro de 8
                                                 --bits auxiliar tenga F0 guardado.
-------------------------------------------------------------------------------------------
error_a <= '1' when (cuentaHL_50us = 2500 and ps2_clk_in = '0') or (ps2_clk_HL = '1' and
cuenta_11fb /= 11 and cuenta_11fb /= 0) else error_r;-- indica un error si el reloj del
                                                     --PS/2 esta a nivel bajo durante
                                                     --mucho tiempo o si este se
                                                     --detiene en el medio de un
                                                     --scancode.
-------------------------------------------------------------------------------------------
tecla <= salida;--asigno a la salida el makecode de la tecla presionada, guardada en el
               --registro de 8 bits de salida.
-------------------------------------------------------------------------------------------
error <= error_a;--asigno a la salida de error la señal de error detectado.
-------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------------------
DET_FLAN_BAJ:process(clk,rst)--Registros del detector de flanco de bajada.
  begin
    if rst = '1' then
      DB <= "00";
    elsif clk'event and clk = '1' then
      DB(0) <= DB(1);
      DB(1) <= ps2_clk_in;
    end if;
  end process DET_FLAN_BAJ;
fb <= DB(1) nor NOT(DB(0));--fb es la salida del detector de flancos de bajada, hay un
                           --flanco de bajada en el reloj del PS/2.
-------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------------------
DET_FLAN_SUB:process(clk,rst)--Registros del detector de flanco de subida.
  begin
    if rst = '1' then
      DS<="11";--asumo que el reloj del PS/2 ha sido elevado durante un tiempo.
    elsif clk'event and clk = '1' then
      DS(0) <= DS(1);
      DS(1) <= ps2_clk_in;
    end if;
  end process DET_FLAN_SUB;
fs <= DS(1) and NOT(DS(0));--fs es la salida del detector de flancos de subida,hay un
                           --flanco de subida en el reloj del PS/2.
-------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------------------
REG_DESP_10:process(clk,rst)--registro de desplazamiento de 10 bits,para recibir en cada
flaco de bajada los 10 bits enviados por el dispositivo(teclado).
  begin
    if rst = '1' then
      RD <= (others => '0');
    elsif clk'event and clk = '1' then
      if fb = '1' then
        RD <= ps2_dato & RD(0 to 8);--concateno el bit de dato del PS/2 con los 9
                                    --primeros bits del registro de desplazamiento.
      end if;
    end if;
  end process REG_DESP_10;
-------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------------------
REG_8_AUX:process(clk,rst)--registro auxiliar de 8 bits,para almacenar los bits de datos
                          --recibidos del dispositivo cada vez que se active la señal
                          --save.
  begin
```

```
    if rst = '1' then
      raux <= (others => '0');
    elsif clk'event and clk = '1' then
      if save = '1' then
        raux <= (RD(2) & RD(3) & RD(4) & RD(5) & RD(6) & RD(7) & RD(8) & RD(9));
--almaceno los 8 bits de dato del registro de desplazamiento hasta que quiera.
      end if;
    end if;
  end process REG_8_AUX;
-------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------------------
REG_8_SAL:process(clk,rst)--registro de 8 bits, para almacenar los bits de datos
                          --recibidos del dispositivo cada vez que se active la señal
                          --guarda.
  begin
    if rst = '1' then
      salida <= (others => '0');
    elsif clk'event and clk = '1' then
      if guarda = '1' then
        salida <= (RD(2) & RD(3) & RD(4) & RD(5) & RD(6) & RD(7) & RD(8) & RD(9));
--almaceno los 8 bits de dato del registro de desplazamiento hasta que quiera.
      end if;
    end if;
  end process REG_8_SAL;
-------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------------------
TEMP_50usHL_ps2_clk:process(clk,rst)--temporizador que mide el tiempo de la señal de
                                    --reloj del PS/2 entre flanco y flanco,de subida y
                                    --bajada. Ademas temporiza los 50us y un ciclo de
                                    --reloj mas, del reloj del host(FPG), despues de un
                                    --flanco de subida.
  begin
    if rst = '1' then
      cuentaHL_50us <= 0;
    elsif clk'event and clk = '1' then
      if fcuentaHL_50us = '1' then
        cuentaHL_50us <= 0;--reinicio el temporizador cuando finaliza la cuenta.
      else
        if flanco = '1' then
          cuentaHL_50us <= 0;--en cada flanco reinicio el contador, ya sea de subida o
                             --bajada.
        else
          cuentaHL_50us <= cuentaHL_50us + 1;--cuento hasta el proximo flanco detectado.
        end if;
      end if;
    end if;
  end process TEMP_50usHL_ps2_clk;
fcuentaHL_50us <= '1' when cuentaHL_50us = 2500 else '0';--señal para reiniciar el
                                                         --temporizador cuando haya
                                                         --alcanzado su limite.
-------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------------------
  CONT_11F_BAJ:process(clk,rst)--contador para ver si se han enviado los 11 bits del
                               --dispositivo al host.
  begin
    if rst = '1' then
      cuenta_11fb <= 0;
    elsif clk'event and clk = '1' then
      if fcuenta_11fb = '1' then
        cuenta_11fb <= 0;--reinicio el contador cuando finaliza la cuenta.
      else
        if fb = '1' then
          cuenta_11fb <= cuenta_11fb + 1;--cuento con cada flanco de bajada.
        else
          if ps2_clk_HL = '1' or error_a = '1' then
            cuenta_11fb <= 0;--reinicio la cuenta si el reloj del PS/2 se detuvo o si
                             --hay algun error.
          end if;
        end if;
      end if;
    end if;
  end process CONT_11F_BAJ;
-------------------------------------------------------------------------------------------
```

```
-------------------------------------------------------------------------------------------
ACT_ERROR:process(rst,clk)--Actualizo la señal de error.
  begin
    if rst = '1' then
      error_r <= '0';--borro cualquier error almacenado.
    elsif clk'event and clk = '1' then
      error_r <= error_a;--mantengo el error del estado anterior.
    end if;
  end process ACT_ERROR;
-----------------------------------------------------------------------
--Maquina de estados para controlar el envio de los posibles scancodes---
--que envia el dispositivo(teclado) al host(FPGA).<MC>...<BC(F0)><MC>----
-----------------------------------------------------------------------
SEQ_FSM:process(clk,rst)--Parte secuencial de la máquina de estados FSM.
  begin
    if rst = '1' then
      FSM_EA <= T_pres;--voy al estado inicial.
    elsif clk'event and clk = '1' then
      FSM_EA <= FSM_ES;--voy al siguiente estado.
    end if;
  end process SEQ_FSM;
-------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------------------
COMB_FSM:process(makecode,breakcode,FSM_EA)--Parte combinacional de la máquina de
                                           --estados FSM.
  begin
    Case FSM_EA is
      when T_pres =>
        guarda       <= '0';
        save         <= '0';
        fcuenta_11fb <= '0';
      if breakcode = '1' then
        FSM_ES       <= T_libe;--si hay un breakcode me voy al estado T_libe.
        fcuenta_11fb <= '1';--reinicio la cuenta de 11 bits.
      else
        if makecode = '1' then
          FSM_ES       <= T_pres;--si no hay breakcode y hay un makecode me voy al
                                 --estado T_pres.
          save         <= '1';--si no hay breakcode y hay un makecode guardo en el
                              --registro auxiliar los 8 bits recibidos en esa trama.
          fcuenta_11fb <= '1';--reinicio la cuenta de 11 bits.
        else
          FSM_ES <= T_pres;--sino me quedo en este estado.
        end if;
      end if;
      when T_libe =>
        guarda       <= '0';
        save         <= '0';
        fcuenta_11fb <= '0';
      if makecode = '1' then
        FSM_ES       <= Imprime_tecla;--si hay un makecode me voy al estado
                                      --Imprime_tecla.
        save         <= '1';--si hay un makecode guardo en el registro auxiliar los 8
                            --bits recibidos en esa trama.
        fcuenta_11fb <= '1';--reinicio la cuenta de 11 bits.
      else
        FSM_ES <= T_libe;--sino me quedo en este estado.
      end if;
      when Imprime_tecla =>
        FSM_ES       <= T_pres;--en este estado pase lo que pase sigo en el próximo
                               --ciclo de reloj al estado inicial.
        guarda       <= '1';--guardo en el registro de salida los 8 bits recibidos en
                            --esa trama.
        save         <= '0';
        fcuenta_11fb <= '0';--reinicio la cuenta de 11 bits.
    end Case;
  end process COMB_FSM;
-------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------------------
end BEH;
```

16.2. Código *VHDL* de la Pantalla *VGA*.

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_ARITH.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;
entity VGA is
  port(clk   : in std_logic;--reloj principal de la fpga (50MHz)->T=20ns.
       rst   : in std_logic;--reinicio global del sistema.
       Color : in std_logic_vector(2 downto 0);--señal de 3 bits que indica el código
                                               --del color a dar al píxel que se esté
                                               --procesando.
       HSync : out std_logic;--señal de sincronismo horizontal activa a nivel bajo de 1
                             --bit, que se activa a los 656 ciclos de reloj.
       VSync : out std_logic;--señal de sincronismo vertical activa a nivel bajo de 1
                             --bit, que se activa a los 416 771 ciclos de reloj.
       R,G,B : out std_logic;--señales de 1 bit que sirven para seleccionar el color a
                             --sacar por pantalla.
       X     : out std_logic_vector(9 downto 0);--señal de 10 bits (2^10=1024) para
                                                --alcanzar los 800 pixeles de ancho.
       Y     : out std_logic_vector(9 downto 0));--señal de 10 bits (2^10=1024) para
                                                 --alcanzar los 521 pixeles de alto.
end VGA;
architecture BEH of VGA is
-------------Declaracion de señales-----------------
signal               t: std_logic;--señal de entrada al FF_T.
signal          Enable: std_logic;--señal de salida al FF_T.
signal           q_act: std_logic;--señal auxiliar para el estado actual del FF_T.
signal           q_sig: std_logic;--señal auxiliar para el estado siguiente del FF_T,
                                  --para trabajar a 25MHz.
signal        cuenta_h: integer range 0 to 800;--señal del contador para el barrido
                                               --horizontal.
signal        cuenta_v: integer range 0 to 521;--señal del contador para el barrido
                                               --vertical.
signal clear_h,clear_v: std_logic;--señales para reiniciar los contadores de barrido,
                                  --tanto horizontal como vertical respectivamente.
signal HBlanck,VBlanck: std_logic;--señales de entrada al bloque blanqueador para
                                  --eliminar los colores en R,G y B.
signal      a_cuenta_h: std_logic_vector(9 downto 0);--señal auxiliar para poder
                                                     --convertir la cuenta a
                                                     --std_logic_vector.
signal      a_cuenta_v: std_logic_vector(9 downto 0);--señal auxiliar para poder
                                                     --convertir la cuenta a
                                                     --std_logic_vector.
Begin
-----------------------------------------------------------------------
--Convierto los enteros de la cuenta de los contadores a un vector--
--logico y poder obtener correctamente la salida.                 --
-----------------------------------------------------------------------
a_cuenta_h <= conv_std_logic_vector(cuenta_h, 10);--convierto cuenta_h a
                                                  --std_logic_vector de 10 bits de
                                                  --longitud.
-------------------------------------------------------------------------------------------
a_cuenta_v <= conv_std_logic_vector(cuenta_v, 10);--convierto cuenta_v a
                                                  --std_logic_vector de 10 bits de
                                                  --longitud.
-------------------------------------------------------------------------------------------
X <= a_cuenta_h;--asigno la cuenta ya en std_logic_vector a la salida.
Y <= a_cuenta_v;--asigno la cuenta ya en std_logic_vector a la salida.
-----------------------------------------------------------------
--Bloque blanqueador que se encarga de eliminar los colores de--
--las salidas R,G y B.                                        --
-----------------------------------------------------------------
R <= '0' when HBlanck = '1' or VBlanck = '1' else Color(2);
G <= '0' when HBlanck = '1' or VBlanck = '1' else Color(1);
B <= '0' when HBlanck = '1' or VBlanck = '1' else Color(0);
-----------------------------------------------------------------
--Biestable tipo T con la funcion de divisor de frecuencia  --
--la placa tiene 50 MHz y los tiempos estan calculados para --
--25MHz.Funcionara como un Enable para el resto del circuito--
-----------------------------------------------------------------
FF_T:process(clk,rst)
  begin
```

```
    if (rst = '1') then
      q_act <= '0';
    elsif (clk'event and clk = '1') then
      q_act <= q_sig;
    end if;
  end process FF_T;
t <= '1';--garantizo que el biestable cambiara de estado cada dos cilos de reloj de
        --50MHz,asegurando los calculos para 25MHz.
q_sig <= q_act when t = '0' else not(q_act);--cambia solo cuando la señal 't' = 1.
Enable <= q_act;--asigno la salida del FF_T.
----------BARRIDO HORIZONTAL (X)------------------------------
-------------------------------------------------------------
--Contador de 10 bits para barrer las lineas horizontales de--
--la pantalla  con resolucion 640x480.                      --
-------------------------------------------------------------
CONT_H:process(clk,rst)
  begin
    if rst = '1' then
      cuenta_h <= 0;
    elsif clk'event and clk = '1' then
      if clear_h = '1' then
        cuenta_h <= 0;
      elsif Enable = '1' then
        cuenta_h <= cuenta_h + 1;
      end if;
    end if;
  end process CONT_H;
-------------------------------------------------------------
--Comparador que genera las señales de sincronismo, blanqueo--
-- y borra el contador de barrido horizontal de la pantalla.--
-------------------------------------------------------------
 HSync   <= '0' when (cuenta_h <= 752) and (cuenta_h >= 656) else '1';
 HBlanck <= '1' when (cuenta_h >= 640) and (cuenta_h <= 800) else '0';
 clear_h <= '1' when cuenta_h = 800 else '0';
----------BARRIDO VERTICAL (Y)--------------------------------
-------------------------------------------------------------
--Contador de 10 bits para barrer las lineas verticales de  --
--la pantalla  con resolucion 640x480.                      --
-------------------------------------------------------------
CONT_V:process(clk,rst)
  begin
    if rst = '1' then
      cuenta_v <= 0;
    elsif clk'event and clk = '1' then
      if clear_v = '1' then
        cuenta_v <= 0;
      elsif clear_h = '1' then
        cuenta_v <= cuenta_v + 1;
      end if;
    end if;
  end process CONT_V;
-------------------------------------------------------------
--Comparador que genera las señales de sincronismo, blanqueo--
-- y borra el contador de barrido horizontal de la pantalla.--
-------------------------------------------------------------
 VSync   <= '0' when (cuenta_v <= 492) and (cuenta_v >= 490) else '1';
 VBlanck <= '1' when (cuenta_v >= 480) and (cuenta_v <= 521) else '0';
 clear_v <= '1' when cuenta_v = 521 else '0';
end BEH;
```

16.3. Código *VHDL* del Controlador o Driver.

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_ARITH.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;
entity Controlador is
  port(clk     : in std_logic;--reloj principal de la fpga (50MHz)->T=20ns.
       rst     : in std_logic;--reinicio global del sistema.
       c_tecla : in std_logic_vector(7 downto 0);--codigo hexadecimal de la tecla
                                                 --presionada.
       c_error : in std_logic;--señal de salida de error.
       c_X     : in std_logic_vector(9 downto 0);--señal de 10 bits (2^10=1024) para
                                                 --alcanzar los 640 pixeles de ancho.
       c_Y     : in std_logic_vector(9 downto 0);--señal de 10 bits (2^10=1024) para
                                                 --alcanzar los 480 pixeles de alto.
       c_color : out std_logic_vector(2 downto 0));--señal de 3 bits que indica el
                                                   --código del Color a dar al píxel que
                                                   --se esté procesando.
end Controlador;
architecture BEH of Controlador is
-------------Declaracion de señales auxiliares-----------------
signal X,Y: integer range 0 to 800;
signal f,c: integer range 0 to 32;
Begin
X <= conv_integer(c_X);--convierto las x a numeros enteros.
Y <= conv_integer(c_Y);--convierto las y a numeros enteros.
-------------------------------------------------------------------------------------
COLUMNAS:process(X)--Proceso para dividir la pantalla en 32 columnas.
  begin
    case X is
      when 0 to 19    => c <= 1;
      when 20 to 39   => c <= 2;
      when 40 to 59   => c <= 3;
      when 60 to 79   => c <= 4;
      when 80 to 99   => c <= 5;
      when 100 to 119 => c <= 6;
      when 120 to 139 => c <= 7;
      when 140 to 159 => c <= 8;
      when 160 to 179 => c <= 9;
      when 180 to 199 => c <= 10;
      when 200 to 219 => c <= 11;
      when 220 to 239 => c <= 12;
      when 240 to 259 => c <= 13;
      when 260 to 279 => c <= 14;
      when 280 to 299 => c <= 15;
      when 300 to 319 => c <= 16;
      when 320 to 339 => c <= 17;
      when 340 to 359 => c <= 18;
      when 360 to 379 => c <= 19;
      when 380 to 399 => c <= 20;
      when 400 to 419 => c <= 21;
      when 420 to 439 => c <= 22;
      when 440 to 459 => c <= 23;
      when 460 to 479 => c <= 24;
      when 480 to 499 => c <= 25;
      when 500 to 519 => c <= 26;
      when 520 to 539 => c <= 27;
      when 540 to 559 => c <= 28;
      when 560 to 579 => c <= 29;
      when 580 to 599 => c <= 30;
      when 600 to 619 => c <= 31;
      when 620 to 639 => c <= 32;
      when others     => c <= 0;
    end case;
  end process COLUMNAS;
-------------------------------------------------------------------------------------
FILAS:process(Y)--Proceso para dividir la pantalla en 24 filas.
  begin
    case Y is
      when 0 to 19    => f <= 1;
      when 20 to 39   => f <= 2;
      when 40 to 59   => f <= 3;
```

```
      when 60 to 79   => f <= 4;
      when 80 to 99   => f <= 5;
      when 100 to 119 => f <= 6;
      when 120 to 139 => f <= 7;
      when 140 to 159 => f <= 8;
      when 160 to 179 => f <= 9;
      when 180 to 199 => f <= 10;
      when 200 to 219 => f <= 11;
      when 220 to 239 => f <= 12;
      when 240 to 259 => f <= 13;
      when 260 to 279 => f <= 14;
      when 280 to 299 => f <= 15;
      when 300 to 319 => f <= 16;
      when 320 to 339 => f <= 17;
      when 340 to 359 => f <= 18;
      when 360 to 379 => f <= 19;
      when 380 to 399 => f <= 20;
      when 400 to 419 => f <= 21;
      when 420 to 439 => f <= 22;
      when 440 to 459 => f <= 23;
      when 460 to 479 => f <= 24;
      when others     => f <= 0;
    end case;
  end process FILAS;
-------------------------------------------------------------------------------------------
PRUEBA:process(c_tecla,c_error,f,c,clk,rst)
begin
  if rst = '1' then
    c_color <= "111";--color blanco.
  elsif c_error = '1' then
    c_color <= "100";--Color rojo.
  elsif clk'event and clk = '1' then
    CASE c_tecla IS
      when "00000101" => --tecla F1([05] patron de pruebas1).
        if f>12 then
          if c>16 then
            c_color <= "100";--d_i
          else
            c_color <= "001";--i_i
          end if;
        else
          if c>16 then
            c_color <= "010";--d_s
          else
            c_color <= "111";--i_s
          end if;
        end if;
      when "00000110" => --tecla F2([06] patron de pruebas2).
        if f>12 then
          if c>16 then
            c_color <= "001";--d_i
          else
            c_color <= "111";--i_i
          end if;
        else
          if c>16 then
            c_color <= "100";--d_s
          else
            c_color <= "010";--i_s
          end if;
        end if;
      when "00000100" => --tecla F3([04] patron de pruebas3).
        if f>12 then
          if c>16 then
            c_color <= "111";--d_i
          else
            c_color <= "010";--i_i
          end if;
        else
          if c>16 then
            c_color <= "001";--d_s
          else
            c_color <= "100";--i_s
```

```
        end if;
      end if;
    when "00001100" => --tecla F4([0C] patron de pruebas4).
      if f>12 then
        if c>16 then
          c_color <= "010";--d_i
        else
          c_color <= "100";--i_i
        end if;
      else
        if c>16 then
          c_color <= "111";--d_s
        else
          c_color <= "001";--i_s
        end if;
      end if;
    when "00000011" => --tecla F5([03] patron de pruebas5).
      if f>12 then
        if c>16 then
          c_color <= "100";--d_i
        else
          c_color <= "001";--i_i
        end if;
      else
        if c>16 then
          c_color <= "010";--d_s
        else
          c_color <= "111";--i_s
        end if;
      end if;
    when "00001011" => --tecla F6([0B] patron de pruebas6).
      if f>12 then
        if c>16 then
          c_color <= "001";--d_i
        else
          c_color <= "111";--i_i
        end if;
      else
        if c>16 then
          c_color <= "100";--d_s
        else
          c_color <= "010";--i_s
        end if;
      end if;
    when "10000011" => --tecla F7([83] patron de pruebas7).
      if f>12 then
        if c>16 then
          c_color <= "111";--d_i
        else
          c_color <= "010";--i_i
        end if;
      else
        if c>16 then
          c_color <= "001";--d_s
        else
          c_color <= "100";--i_s
        end if;
      end if;
    when "00001010" => --tecla F8([0A] patron de pruebas8).
      if f>12 then
        if c>16 then
          c_color <= "010";--d_i
        else
          c_color <= "100";--i_i
        end if;
      else
        if c>16 then
          c_color <= "111";--d_s
        else
          c_color <= "001";--i_s
        end if;
      end if;
    when "00000001" => --tecla F9([01] patron de pruebas9).
```

```
        if f>12 then
          if c>16 then
            c_color <= "100";--d_i
          else
            c_color <= "001";--i_i
          end if;
        else
          if c>16 then
            c_color <= "010";--d_s
          else
            c_color <= "111";--i_s
          end if;
        end if;
      when "00001001" => --tecla F10([09] patron de pruebas10).
        if f>12 then
          if c>16 then
            c_color <= "001";--d_i
          else
            c_color <= "111";--i_i
          end if;
        else
          if c>16 then
            c_color <= "100";--d_s
          else
            c_color <= "010";--i_s
          end if;
        end if;
      when "01111000" => --tecla F11([78] patron de pruebas11).
        if f>12 then
          if c>16 then
            c_color <= "111";--d_i
          else
            c_color <= "010";--i_i
          end if;
        else
          if c>16 then
            c_color <= "001";--d_s
          else
            c_color <= "100";--i_s
          end if;
        end if;
      when "00000111" => --tecla F12([07] patron de pruebas12).
        if f>12 then
          if c>16 then
            c_color <= "010";--d_i
          else
            c_color <= "100";--i_i
          end if;
        else
          if c>16 then
            c_color <= "111";--d_s
          else
            c_color <= "001";--i_s
          end if;
        end if;
      when "01011010" => --tecla ENTER([5A]).
        if f<=6 then
          c_color <= "111";
        else
          if f=7 or f=17 then
            if c>=11 and c<=21 then
              c_color <= "100";
            else
              c_color <= "111";
            end if;
          else
            if f=8 or f=16 then
              if c=11 or c=21 then
                c_color <= "100";
              else
                c_color <= "111";
              end if;
            else
```

```
            if f=9 or f=10 then
              if c=11 or c=21 then
                c_color <= "100";
              else
                if c=18 then
                  c_color <= "000";
                else
                  c_color <= "111";
                end if;
              end if;
            else
              if f=11 then
                if c=11 or c=21 then
                  c_color <= "100";
                else
                  if c=16 or c=18 then
                    c_color <= "000";
                  else
                    c_color <= "111";
                  end if;
                end if;
              else
                if f=12 then
                  if c=11 or c=21 then
                    c_color <= "100";
                  else
                    if c=15 or c=16 or c=18 then
                      c_color <= "000";
                    else
                      c_color <= "111";
                    end if;
                  end if;
                else
                  if f=13 then
                    if c=11 or c=21 then
                      c_color <= "100";
                    else
                      if c>=14 and c<=18 then
                        c_color <= "000";
                      else
                        c_color <= "111";
                      end if;
                    end if;
                  else
                    if f=14 then
                      if c=11 or c=21 then
                        c_color <= "100";
                      else
                        if c=15 or c=16 then
                          c_color <= "000";
                        else
                          c_color <= "111";
                        end if;
                      end if;
                    else
                      if f=15 then
                        if c=11 or c=21 then
                          c_color <= "100";
                        else
                          if c=16 then
                            c_color <= "000";
                          else
                            c_color <= "111";
                          end if;
                        end if;
                      else
                        if f>17 then
                          c_color <= "111";
                        end if;
                      end if;
                    end if;
                  end if;
                end if;
              end if;
```

```
            end if;
          end if;
        end if;
      end if;
    end if;
  when "00101001" => --tecla SPACE([29]).
    if f<=6 then
      c_color <= "111";
    else
      if f=7 or f=17 then
        if c>=11 and c<=21 then
          c_color <= "100";
        else
          c_color <= "111";
        end if;
      else
        if f=8 or f=9 or f=10 or f=11 or f=14 or f=15 or f=16 then
          if c=11 or c=21 then
            c_color <= "100";
          else
            c_color <= "111";
          end if;
        else
          if f=12 then
            if c=11 or c=21 then
              c_color <= "100";
            else
              if c=14 or c=18 then
                c_color <= "000";
              else
                c_color <= "111";
              end if;
            end if;
          else
            if f=13 then
              if c=11 or c=21 then
                c_color <= "100";
              else
                if c>=14 and c<=18 then
                  c_color <= "000";
                else
                  c_color <= "111";
                end if;
              end if;
            else
              if f>17 then
                c_color <= "111";
              end if;
            end if;
          end if;
        end if;
      end if;
    end if;
  when "00010100" => --tecla CONTROL([14]).
    if f<=6 then
      c_color <= "111";
    else
      if f=7 or f=17 then
        if c>=11 and c<=21 then
          c_color <= "100";
        else
          c_color <= "111";
        end if;
      else
        if f=8 or f=9 or f=15 or f=16 then
          if c=11 or c=21 then
            c_color <= "100";
          else
            c_color <= "111";
          end if;
        else
          if f=10 then
            if c=11 or c=21 then
```

```
                c_color <= "100";
              else
                if c=14 or c=15 or c=17 then
                  c_color <= "000";
                else
                  c_color <= "111";
                end if;
              end if;
            else
              if f=11 then
                if c=11 or c=21 then
                  c_color <= "100";
                else
                  if c=14 or c=17 or c=18 then
                    c_color <= "000";
                  else
                    c_color <= "111";
                  end if;
                end if;
              else
                if f=12 or f=13 then
                  if c=11 or c=21 then
                    c_color <= "100";
                  else
                    if c=14 or c=17 then
                      c_color <= "000";
                    else
                      c_color <= "111";
                    end if;
                  end if;
                else
                  if f=14 then
                    if c=11 or c=21 then
                      c_color <= "100";
                    else
                      if c=14 or c=15 or c=17 or c=18 then
                        c_color <= "000";
                      else
                        c_color <= "111";
                      end if;
                    end if;
                  else
                    if f>17 then
                      c_color <= "111";
                    end if;
                  end if;
                end if;
              end if;
            end if;
          end if;
        end if;
      end if;
    when "00010001" => --tecla ALT([11]).
      if f<=6 then
        c_color <= "111";
      else
        if f=7 or f=17 then
          if c>=11 and c<=21 then
            c_color <= "100";
          else
            c_color <= "111";
          end if;
        else
          if f=8 or f=9 or f=15 or f=16 then
            if c=11 or c=21 then
              c_color <= "100";
            else
              c_color <= "111";
            end if;
          else
            if f=10 then
              if c=11 or c=21 then
                c_color <= "100";
```

```
              else
                if c=15 or c=18 then
                  c_color <= "000";
                else
                  c_color <= "111";
                end if;
              end if;
            else
              if f=11 or f=13 or f=14 then
                if c=11 or c=21 then
                  c_color <= "100";
                else
                  if c=14 or c=16 or c=18 then
                    c_color <= "000";
                  else
                    c_color <= "111";
                  end if;
                end if;
              else
                if f=12 then
                  if c=11 or c=21 then
                    c_color <= "100";
                  else
                    if c=14 or c=15 or c=16 or c=18 then
                      c_color <= "000";
                    else
                      c_color <= "111";
                    end if;
                  end if;
                else
                  if f>17 then
                    c_color <= "111";
                  end if;
                end if;
              end if;
            end if;
          end if;
        end if;
      when "00001101" => --tecla TAB([0D]).
        if f<=6 then
          c_color <= "111";
        else
          if f=7 or f=17 then
            if c>=11 and c<=21 then
              c_color <= "100";
            else
              c_color <= "111";
            end if;
          else
            if f=8 or f=12 or f=16 then
              if c=11 or c=21 then
                c_color <= "100";
              else
                c_color <= "111";
              end if;
            else
              if f=9 or f=11 then
                if c=21 or c=11 then
                  c_color <= "100";
                else
                  if c=14 then
                    c_color <= "000";
                  else
                    c_color <= "111";
                  end if;
                end if;
              else
                if f=10 or f=14 then
                  if c=21 or c=11 then
                    c_color <= "100";
                  else
                    if c>=14 and c<=18 then
```

```
                    c_color <= "000";
                  else
                    c_color <= "111";
                  end if;
                end if;
              else
                if f=13 or f=15 then
                  if c=21 or c=11 then
                    c_color <= "100";
                  else
                    if c=18 then
                      c_color <= "000";
                    else
                      c_color <= "111";
                    end if;
                  end if;
                else
                  if f>17 then
                    c_color <= "111";
                  end if;
                end if;
              end if;
            end if;
          end if;
        end if;
      when "00010010" => --tecla L SHIFT([12]).
        if f<=6 then
          c_color <= "111";
        else
          if f=7 or f=17 then
            if c>=11 and c<=21 then
              c_color <= "100";
            else
              c_color <= "111";
            end if;
          else
            if f=8 or f=16 then
              if c=11 or c=21 then
                c_color <= "100";
              else
                c_color <= "111";
              end if;
            else
              if f=9 then
                if c=21 or c=11 then
                  c_color <= "100";
                else
                  if c=16 then
                    c_color <= "000";
                  else
                    c_color <= "111";
                  end if;
                end if;
              else
                if f=10 then
                  if c=21 or c=11 then
                    c_color <= "100";
                  else
                    if c>=15 and c<=17 then
                      c_color <= "000";
                    else
                      c_color <= "111";
                    end if;
                  end if;
                else
                  if f=11 then
                    if c=21 or c=11 then
                      c_color <= "100";
                    else
                      if c>=14 and c<=18 then
                        c_color <= "000";
                      else
```

```
                    c_color <= "111";
                  end if;
                end if;
              else
                if f=12 or f=13 or f=14 or f=15 then
                  if c=21 or c=11 then
                    c_color <= "100";
                  else
                    if c=16 then
                      c_color <= "000";
                    else
                      c_color <= "111";
                    end if;
                  end if;
                else
                  if f>17 then
                    c_color <= "111";
                  end if;
                end if;
              end if;
            end if;
          end if;
        end if;
      end if;
    when "01011001" => --tecla R SHIFT([59]).
      if f<=6 then
        c_color <= "111";
      else
        if f=7 or f=17 then
          if c>=11 and c<=21 then
            c_color <= "100";
          else
            c_color <= "111";
          end if;
        else
          if f=8 or f=16 then
            if c=11 or c=21 then
              c_color <= "100";
            else
              c_color <= "111";
            end if;
          else
            if f=9 then
              if c=21 or c=11 then
                c_color <= "100";
              else
                if c=16 then
                  c_color <= "000";
                else
                  c_color <= "111";
                end if;
              end if;
            else
              if f=10 then
                if c=21 or c=11 then
                  c_color <= "100";
                else
                  if c>=15 and c<=17 then
                    c_color <= "000";
                  else
                    c_color <= "111";
                  end if;
                end if;
              else
                if f=11 then
                  if c=21 or c=11 then
                    c_color <= "100";
                  else
                    if c>=14 and c<=18 then
                      c_color <= "000";
                    else
                      c_color <= "111";
```

```
                  end if;
                end if;
              else
                if f=12 or f=13 or f=14 or f=15 then
                  if c=21 or c=11 then
                    c_color <= "100";
                  else
                    if c=16 then
                      c_color <= "000";
                    else
                      c_color <= "111";
                    end if;
                  end if;
                else
                  if f>17 then
                    c_color <= "111";
                  end if;
                end if;
              end if;
            end if;
          end if;
        end if;
      end if;
    when "00011111" => --tecla L GUI([1F]).
      if f<=6 then
        c_color <= "111";
      else
        if f=7 or f=17 then
          if c>=11 and c<=21 then
            c_color <= "100";
          else
            c_color <= "111";
          end if;
        else
          if f=8 or f=12 or f=16 then
            if c=11 or c=21 then
              c_color <= "100";
            else
              c_color <= "111";
            end if;
          else
            if f=9 or f=10 or f=11 or f=13 or f=14 or f=15 then
              if c=21 or c=11 then
                c_color <= "100";
              else
                if c=14 or c=15 or c=17 or c=18 then
                  c_color <= "000";
                else
                  c_color <= "111";
                end if;
              end if;
            else
              if f>17 then
                c_color <= "111";
              end if;
            end if;
          end if;
        end if;
      end if;
    when "00100111" => --tecla R GUI([27]).
      if f<=6 then
        c_color <= "111";
      else
        if f=7 or f=17 then
          if c>=11 and c<=21 then
            c_color <= "100";
          else
            c_color <= "111";
          end if;
        else
          if f=8 or f=12 or f=16 then
            if c=11 or c=21 then
```

```
              c_color <= "100";
            else
              c_color <= "111";
            end if;
          else
            if f=9 or f=10 or f=11 or f=13 or f=14 or f=15 then
              if c=21 or c=11 then
                c_color <= "100";
              else
                if c=14 or c=15 or c=17 or c=18 then
                  c_color <= "000";
                else
                  c_color <= "111";
                end if;
              end if;
            else
              if f>17 then
                c_color <= "111";
              end if;
            end if;
          end if;
        end if;
      end if;
    when "01011000" => --tecla CAPS([58]).
      if f<=6 then
        c_color <= "111";
      else
        if f=7 or f=17 then
          if c>=11 and c<=21 then
            c_color <= "100";
          else
            c_color <= "111";
          end if;
        else
          if f=8 or f=14 or f=16 then
            if c=11 or c=21 then
              c_color <= "100";
            else
              c_color <= "111";
            end if;
          else
            if f=9 then
              if c=21 or c=11 then
                c_color <= "100";
              else
                if c=16 then
                  c_color <= "000";
                else
                  c_color <= "111";
                end if;
              end if;
            else
              if f=10 then
                if c=21 or c=11 then
                  c_color <= "100";
                else
                  if c>=15 and c<=17 then
                    c_color <= "000";
                  else
                    c_color <= "111";
                  end if;
                end if;
              else
                if f=11 or f=15 then
                  if c=21 or c=11 then
                    c_color <= "100";
                  else
                    if c>=14 and c<=18 then
                      c_color <= "000";
                    else
                      c_color <= "111";
                    end if;
                  end if;
```

```
                else
                  if f=12 or f=13 then
                    if c=21 or c=11 then
                      c_color <= "100";
                    else
                      if c=16 then
                        c_color <= "000";
                      else
                        c_color <= "111";
                      end if;
                    end if;
                  else
                    if f>17 then
                      c_color <= "111";
                    end if;
                  end if;
                end if;
              end if;
            end if;
          end if;
        end if;
      end if;
    when "01100110" => --tecla BKSP([66]).
      if f<=6 then
        c_color <= "111";
      else
        if f=7 or f=17 then
          if c>=11 and c<=21 then
            c_color <= "100";
          else
            c_color <= "111";
          end if;
        else
          if f=8 or f=9 or f=15 or f=16 then
            if c=11 or c=21 then
              c_color <= "100";
            else
              c_color <= "111";
            end if;
          else
            if f=10 or f=14 then
              if c=21 or c=11 then
                c_color <= "100";
              else
                if c=16 then
                  c_color <= "000";
                else
                  c_color <= "111";
                end if;
              end if;
            else
              if f=11 or f=13 then
                if c=21 or c=11 then
                  c_color <= "100";
                else
                  if c=15 or c=16 then
                    c_color <= "000";
                  else
                    c_color <= "111";
                  end if;
                end if;
              else
                if f=12 then
                  if c=21 or c=11 then
                    c_color <= "100";
                  else
                    if c>=14 and c<=18 then
                      c_color <= "000";
                    else
                      c_color <= "111";
                    end if;
                  end if;
                else
```

```
                if f>17 then
                  c_color <= "111";
                end if;
              end if;
            end if;
          end if;
        end if;
      end if;
    end if;
  when "00101111" => --tecla APPS([2F]).
    if f<=6 then
      c_color <= "111";
    else
      if f=7 or f=17 then
        if c>=11 and c<=21 then
          c_color <= "100";
        else
          c_color <= "111";
        end if;
      else
        if f=8 or f=10 or f=12 or f=14 or f=16 then
          if c=11 or c=21 then
            c_color <= "100";
          else
            c_color <= "111";
          end if;
        else
          if f=9 or f=11 or f=13 or f=15 then
            if c=21 or c=11 then
              c_color <= "100";
            else
              if c>=14 and c<=18 then
                c_color <= "000";
              else
                c_color <= "111";
              end if;
            end if;
          else
            if f>17 then
              c_color <= "111";
            end if;
          end if;
        end if;
      end if;
    end if;
  when "01110110" => --tecla ESC([76]).
    if f<=6 then
      c_color <= "111";
    else
      if f=7 or f=17 then
        if c>=11 and c<=21 then
          c_color <= "100";
        else
          c_color <= "111";
        end if;
      else
        if f=8 or f=9 or f=15 or f=16 then
          if c=11 or c=21 then
            c_color <= "100";
          else
            c_color <= "111";
          end if;
        else
          if f=10 then
            if c=21 or c=11 then
              c_color <= "100";
            else
              if c>=14 and c<=16 then
                c_color <= "000";
              else
                c_color <= "111";
              end if;
            end if;
```

```
            else
              if f=11 then
                if c=21 or c=11 then
                  c_color <= "100";
                else
                  if c=14 or c=15 then
                    c_color <= "000";
                  else
                    c_color <= "111";
                  end if;
                end if;
              else
                if f=12 then
                  if c=21 or c=11 then
                    c_color <= "100";
                  else
                    if c=14 or c=16 then
                      c_color <= "000";
                    else
                      c_color <= "111";
                    end if;
                  end if;
                else
                  if f=13 then
                    if c=21 or c=11 then
                      c_color <= "100";
                    else
                      if c=17 then
                        c_color <= "000";
                      else
                        c_color <= "111";
                      end if;
                    end if;
                  else
                    if f=14 then
                      if c=21 or c=11 then
                        c_color <= "100";
                      else
                        if c=18 then
                          c_color <= "000";
                        else
                          c_color <= "111";
                        end if;
                      end if;
                    else
                      if f>17 then
                        c_color <= "111";
                      end if;
                    end if;
                  end if;
                end if;
              end if;
            end if;
          end if;
        end if;
      when "01110101" => --tecla U ARROW([75]).
        if f<=6 then
          c_color <= "111";
        else
          if f=7 or f=17 then
            if c>=11 and c<=21 then
              c_color <= "100";
            else
              c_color <= "111";
            end if;
          else
            if f=8 or f=9 or f=10 or f=14 or f=15 or f=16 then
              if c=11 or c=21 then
                c_color <= "100";
              else
                c_color <= "111";
              end if;
```

```
            else
              if f=11 then
                if c=11 or c=21 then
                  c_color <= "100";
                else
                  if c=16 then
                    c_color <= "000";
                  else
                    c_color <= "111";
                  end if;
                end if;
              else
                if f=12 then
                  if c=11 or c=21 then
                    c_color <= "100";
                  else
                    if c>=15 and c<=17 then
                      c_color <= "000";
                    else
                      c_color <= "111";
                    end if;
                  end if;
                else
                  if f=13 then
                    if c=11 or c=21 then
                      c_color <= "100";
                    else
                      if c>=14 and c<=18 then
                        c_color <= "000";
                      else
                        c_color <= "111";
                      end if;
                    end if;
                  else
                    if f>17 then
                      c_color <= "111";
                    end if;
                  end if;
                end if;
              end if;
            end if;
          end if;
        end if;
      when "01101011" => --tecla L ARROW([6B]).
        if f<=6 then
          c_color <= "111";
        else
          if f=7 or f=17 then
            if c>=11 and c<=21 then
              c_color <= "100";
            else
              c_color <= "111";
            end if;
          else
            if f=8 or f=9 or f=15 or f=16 then
              if c=11 or c=21 then
                c_color <= "100";
              else
                c_color <= "111";
              end if;
            else
              if f=10 or f=14 then
                if c=11 or c=21 then
                  c_color <= "100";
                else
                  if c=17 then
                    c_color <= "000";
                  else
                    c_color <= "111";
                  end if;
                end if;
              else
                if f=11 or f=13 then
```

```
                if c=11 or c=21 then
                  c_color <= "100";
                else
                  if c=16 or c=17 then
                    c_color <= "000";
                  else
                    c_color <= "111";
                  end if;
                end if;
              else
                if f=12 then
                  if c=11 or c=21 then
                    c_color <= "100";
                  else
                    if c>=15 and c<=17 then
                      c_color <= "000";
                    else
                      c_color <= "111";
                    end if;
                  end if;
                else
                  if f>17 then
                    c_color <= "111";
                  end if;
                end if;
              end if;
            end if;
          end if;
        end if;
      when "01110010" => --tecla D ARROW([72]).
        if f<=6 then
          c_color <= "111";
        else
          if f=7 or f=17 then
            if c>=11 and c<=21 then
              c_color <= "100";
            else
              c_color <= "111";
            end if;
          else
            if f=8 or f=9 or f=10 or f=14 or f=15 or f=16 then
              if c=11 or c=21 then
                c_color <= "100";
              else
                c_color <= "111";
              end if;
            else
              if f=11 then
                if c=11 or c=21 then
                  c_color <= "100";
                else
                  if c>=14 and c<=18 then
                    c_color <= "000";
                  else
                    c_color <= "111";
                  end if;
                end if;
              else
                if f=12 then
                  if c=11 or c=21 then
                    c_color <= "100";
                  else
                    if c>=15 and c<=17 then
                      c_color <= "000";
                    else
                      c_color <= "111";
                    end if;
                  end if;
                else
                  if f=13 then
                    if c=11 or c=21 then
                      c_color <= "100";
```

```
                else
                  if c=16 then
                    c_color <= "000";
                  else
                    c_color <= "111";
                  end if;
                end if;
              else
                if f>17 then
                  c_color <= "111";
                end if;
              end if;
            end if;
          end if;
        end if;
      end if;
    end if;
  when "01110100" => --tecla R ARROW([74]).
    if f<=6 then
      c_color <= "111";
    else
      if f=7 or f=17 then
        if c>=11 and c<=21 then
          c_color <= "100";
        else
          c_color <= "111";
        end if;
      else
        if f=8 or f=9 or f=15 or f=16 then
          if c=11 or c=21 then
            c_color <= "100";
          else
            c_color <= "111";
          end if;
        else
          if f=10 or f=14 then
            if c=11 or c=21 then
              c_color <= "100";
            else
              if c=15 then
                c_color <= "000";
              else
                c_color <= "111";
              end if;
            end if;
          else
            if f=11 or f=13 then
              if c=11 or c=21 then
                c_color <= "100";
              else
                if c=16 or c=15 then
                  c_color <= "000";
                else
                  c_color <= "111";
                end if;
              end if;
            else
              if f=12 then
                if c=11 or c=21 then
                  c_color <= "100";
                else
                  if c>=15 and c<=17 then
                    c_color <= "000";
                  else
                    c_color <= "111";
                  end if;
                end if;
              else
                if f>17 then
                  c_color <= "111";
                end if;
              end if;
            end if;
```

```
            end if;
          end if;
        end if;
      end if;
    when "00001110" => --tecla º ([0E]).
      if f<=6 then
        c_color <= "111";
      else
        if f=7 or f=17 then
          if c>=11 and c<=21 then
            c_color <= "100";
          else
            c_color <= "111";
          end if;
        else
          if f=8 or f=12 or f=13 or f=14 or f=15 or f=16 then
            if c=11 or c=21 then
              c_color <= "100";
            else
              c_color <= "111";
            end if;
          else
            if f=9 or f=11 then
              if c=11 or c=21 then
                c_color <= "100";
              else
                if c>=14 and c<=16 then
                  c_color <= "000";
                else
                  c_color <= "111";
                end if;
              end if;
            else
              if f=10 then
                if c=11 or c=21 then
                  c_color <= "100";
                else
                  if c=14 or c=16 then
                    c_color <= "000";
                  else
                    c_color <= "111";
                  end if;
                end if;
              else
                if f>17 then
                  c_color <= "111";
                end if;
              end if;
            end if;
          end if;
        end if;
      end if;
    when "01001110" => --tecla ' ([4E]).
      if f<=6 then
        c_color <= "111";
      else
        if f=7 or f=17 then
          if c>=11 and c<=21 then
            c_color <= "100";
          else
            c_color <= "111";
          end if;
        else
          if f=8 or f=11 or f=12 or f=13 or f=14 or f=15 or f=16 then
            if c=11 or c=21 then
              c_color <= "100";
            else
              c_color <= "111";
            end if;
          else
            if f=9 or f=10 then
              if c=11 or c=21 then
                c_color <= "100";
```

```
              else
                if c=17 then
                  c_color <= "000";
                else
                  c_color <= "111";
                end if;
              end if;
            else
              if f>17 then
                c_color <= "111";
              end if;
            end if;
          end if;
        end if;
      end if;
    when "01010101" => --tecla ¡ ([55]).
      if f<=6 then
        c_color <= "111";
      else
        if f=7 or f=17 then
          if c>=11 and c<=21 then
            c_color <= "100";
          else
            c_color <= "111";
          end if;
        else
          if f=8 or f=10 or f=16 then
            if c=11 or c=21 then
              c_color <= "100";
            else
              c_color <= "111";
            end if;
          else
            if f=9 or f=11 or f=12 or f=13 or f=14 or f=15 then
              if c=11 or c=21 then
                c_color <= "100";
              else
                if c=16 then
                  c_color <= "000";
                else
                  c_color <= "111";
                end if;
              end if;
            else
              if f>17 then
                c_color <= "111";
              end if;
            end if;
          end if;
        end if;
      end if;
    when "01011101" => --tecla ç ([5D]).
      if f<=6 then
        c_color <= "111";
      else
        if f=7 or f=17 then
          if c>=11 and c<=21 then
            c_color <= "100";
          else
            c_color <= "111";
          end if;
        else
          if f=8 or f=16 then
            if c=11 or c=21 then
              c_color <= "100";
            else
              c_color <= "111";
            end if;
          else
            if f=9 or f=13 then
              if c=11 or c=21 then
                c_color <= "100";
              else
```

```
                if c>=15 and c<=17 then
                  c_color <= "000";
                else
                  c_color <= "111";
                end if;
              end if;
            else
              if f=10 or f=12 then
                if c=11 or c=21 then
                  c_color <= "100";
                else
                  if c=14 or c=18 then
                    c_color <= "000";
                  else
                    c_color <= "111";
                  end if;
                end if;
              else
                if f=11 then
                  if c=11 or c=21 then
                    c_color <= "100";
                  else
                    if c=14 then
                      c_color <= "000";
                    else
                      c_color <= "111";
                    end if;
                  end if;
                else
                  if f=14 then
                    if c=11 or c=21 then
                      c_color <= "100";
                    else
                      if c=16 then
                        c_color <= "000";
                      else
                        c_color <= "111";
                      end if;
                    end if;
                  else
                    if f=15 then
                      if c=11 or c=21 then
                        c_color <= "100";
                      else
                        if c=15 then
                          c_color <= "000";
                        else
                          c_color <= "111";
                        end if;
                      end if;
                    else
                      if f>17 then
                        c_color <= "111";
                      end if;
                    end if;
                  end if;
                end if;
              end if;
            end if;
          end if;
        end if;
      end if;
    when "01010010" => --tecla ´ ([52]).
      if f<=6 then
        c_color <= "111";
      else
        if f=7 or f=17 then
          if c>=11 and c<=21 then
            c_color <= "100";
          else
            c_color <= "111";
          end if;
        else
```

```
            if f=8 or f=11 or f=12 or f=13 or f=14 or f=15 or f=16 then
              if c=11 or c=21 then
                c_color <= "100";
              else
                c_color <= "111";
              end if;
            else
              if f=9 then
                if c=11 or c=21 then
                  c_color <= "100";
                else
                  if c=17 then
                    c_color <= "000";
                  else
                    c_color <= "111";
                  end if;
                end if;
              else
                if f=10 then
                  if c=11 or c=21 then
                    c_color <= "100";
                  else
                    if c=16 then
                      c_color <= "000";
                    else
                      c_color <= "111";
                    end if;
                  end if;
                else
                  if f>17 then
                    c_color <= "111";
                  end if;
                end if;
              end if;
            end if;
          end if;
        end if;
      when "01000001" => --tecla , ([41]).
        if f<=6 then
          c_color <= "111";
        else
          if f=7 or f=17 then
            if c>=11 and c<=21 then
              c_color <= "100";
            else
              c_color <= "111";
            end if;
          else
            if f=8 or f=9 or f=10 or f=11 or f=12 or f=16 then
              if c=11 or c=21 then
                c_color <= "100";
              else
                c_color <= "111";
              end if;
            else
              if f=13 or f=14 then
                if c=11 or c=21 then
                  c_color <= "100";
                else
                  if c=18 then
                    c_color <= "000";
                  else
                    c_color <= "111";
                  end if;
                end if;
              else
                if f=15 then
                  if c=11 or c=21 then
                    c_color <= "100";
                  else
                    if c=17 then
                      c_color <= "000";
                    else
```

```
                    c_color <= "111";
                  end if;
                end if;
              else
                if f>17 then
                  c_color <= "111";
                end if;
              end if;
            end if;
          end if;
        end if;
      end if;
    when "01001001" => --tecla . ([49]).
      if f<=6 then
        c_color <= "111";
      else
        if f=7 or f=17 then
          if c>=11 and c<=21 then
            c_color <= "100";
          else
            c_color <= "111";
          end if;
        else
          if f=8 or f=9 or f=10 or f=11 or f=12 or f=13 or f=14 or f=16 then
            if c=11 or c=21 then
              c_color <= "100";
            else
              c_color <= "111";
            end if;
          else
            if f=15 then
              if c=11 or c=21 then
                c_color <= "100";
              else
                if c=18 then
                  c_color <= "000";
                else
                  c_color <= "111";
                end if;
              end if;
            else
              if f>17 then
                c_color <= "111";
              end if;
            end if;
          end if;
        end if;
      end if;
    when "01110001" => --tecla KP .([71]).
      if f<=6 then
        c_color <= "111";
      else
        if f=7 or f=17 then
          if c>=11 and c<=21 then
            c_color <= "100";
          else
            c_color <= "111";
          end if;
        else
          if f=8 or f=9 or f=10 or f=11 or f=12 or f=13 or f=14 or f=16 then
            if c=11 or c=21 then
              c_color <= "100";
            else
              c_color <= "111";
            end if;
          else
            if f=15 then
              if c=11 or c=21 then
                c_color <= "100";
              else
                if c=18 then
                  c_color <= "000";
                else
```

```
                c_color <= "111";
              end if;
            end if;
          else
            if f>17 then
              c_color <= "111";
            end if;
          end if;
        end if;
      end if;
    end if;
  when "01100001" => --tecla <([61]).
    if f<=6 then
      c_color <= "111";
    else
      if f=7 or f=17 then
        if c>=11 and c<=21 then
          c_color <= "100";
        else
          c_color <= "111";
        end if;
      else
        if f=8 or f=16 then
          if c=11 or c=21 then
            c_color <= "100";
          else
            c_color <= "111";
          end if;
        else
          if f=9 or f=15 then
            if c=11 or c=21 then
              c_color <= "100";
            else
              if c=17 then
                c_color <= "000";
              else
                c_color <= "111";
              end if;
            end if;
          else
            if f=10 or f=14 then
              if c=11 or c=21 then
                c_color <= "100";
              else
                if c=16 then
                  c_color <= "000";
                else
                  c_color <= "111";
                end if;
              end if;
            else
              if f=11 or f=13 then
                if c=11 or c=21 then
                  c_color <= "100";
                else
                  if c=15 then
                    c_color <= "000";
                  else
                    c_color <= "111";
                  end if;
                end if;
              else
                if f=12 then
                  if c=11 or c=21 then
                    c_color <= "100";
                  else
                    if c=14 then
                      c_color <= "000";
                    else
                      c_color <= "111";
                    end if;
                  end if;
                else
```

```
                  if f>17 then
                    c_color <= "111";
                  end if;
                end if;
              end if;
            end if;
          end if;
        end if;
      end if;
    end if;
  when "01001010" => --tecla - o KP /([4A]).
    if f<=6 then
      c_color <= "111";
    else
      if f=7 or f=17 then
        if c>=11 and c<=21 then
          c_color <= "100";
        else
          c_color <= "111";
        end if;
      else
        if f=8 or f=9 or f=10 or f=11 or f=13 or f=14 or f=15 or f=16 then
          if c=11 or c=21 then
            c_color <= "100";
          else
            c_color <= "111";
          end if;
        else
          if f=12 then
            if c=11 or c=21 then
              c_color <= "100";
            else
              if c>=15 and c<=17 then
                c_color <= "000";
              else
                c_color <= "111";
              end if;
            end if;
          else
            if f>17 then
              c_color <= "111";
            end if;
          end if;
        end if;
      end if;
    end if;
  when "01010100" => --tecla ` ([54]).
    if f<=6 then
      c_color <= "111";
    else
      if f=7 or f=17 then
        if c>=11 and c<=21 then
          c_color <= "100";
        else
          c_color <= "111";
        end if;
      else
        if f=8 or f=11 or f=12 or f=13 or f=14 or f=15 or f=16 then
          if c=11 or c=21 then
            c_color <= "100";
          else
            c_color <= "111";
          end if;
        else
          if f=9 then
            if c=11 or c=21 then
              c_color <= "100";
            else
              if c=15 then
                c_color <= "000";
              else
                c_color <= "111";
              end if;
```

```
              end if;
            else
              if f=10 then
                if c=11 or c=21 then
                  c_color <= "100";
                else
                  if c=16 then
                    c_color <= "000";
                  else
                    c_color <= "111";
                  end if;
                end if;
              else
                if f>17 then
                  c_color <= "111";
                end if;
              end if;
            end if;
          end if;
        end if;
      when "01011011" => --tecla + ([5B]).
        if f<=6 then
          c_color <= "111";
        else
          if f=7 or f=17 then
            if c>=11 and c<=21 then
              c_color <= "100";
            else
              c_color <= "111";
            end if;
          else
            if f=8 or f=9 or f=15 or f=16 then
              if c=11 or c=21 then
                c_color <= "100";
              else
                c_color <= "111";
              end if;
            else
              if f=10 or f=11 or f=13 or f=14 then
                if c=11 or c=21 then
                  c_color <= "100";
                else
                  if c=16 then
                    c_color <= "000";
                  else
                    c_color <= "111";
                  end if;
                end if;
              else
                if f=12 then
                  if c=11 or c=21 then
                    c_color <= "100";
                  else
                    if c>=14 and c<=18 then
                      c_color <= "000";
                    else
                      c_color <= "111";
                    end if;
                  end if;
                else
                  if f>17 then
                    c_color <= "111";
                  end if;
                end if;
              end if;
            end if;
          end if;
        end if;
      when "01111100" => --tecla KP * ([7C]).
        if f<=6 then
          c_color <= "111";
        else
```

```
          if f=7 or f=17 then
            if c>=11 and c<=21 then
              c_color <= "100";
            else
              c_color <= "111";
            end if;
          else
            if f=8 or f=9 or f=15 or f=16 then
              if c=11 or c=21 then
                c_color <= "100";
              else
                c_color <= "111";
              end if;
            else
              if f=10 or f=14 then
                if c=11 or c=21 then
                  c_color <= "100";
                else
                  if c=14 or c=16 or c=18 then
                    c_color <= "000";
                  else
                    c_color <= "111";
                  end if;
                end if;
              else
                if f=11 or f=13 then
                  if c=11 or c=21 then
                    c_color <= "100";
                  else
                    if c>=15 and c<=17 then
                      c_color <= "000";
                    else
                      c_color <= "111";
                    end if;
                  end if;
                else
                  if f=12 then
                    if c=11 or c=21 then
                      c_color <= "100";
                    else
                      if c>=14 and c<=18 then
                        c_color <= "000";
                      else
                        c_color <= "111";
                      end if;
                    end if;
                  else
                    if f>17 then
                      c_color <= "111";
                    end if;
                  end if;
                end if;
              end if;
            end if;
          end if;
        end if;
      when "01110111" => --tecla NUM ([77]).
        if f<=6 then
          c_color <= "111";
        else
          if f=7 or f=17 then
            if c>=11 and c<=21 then
              c_color <= "100";
            else
              c_color <= "111";
            end if;
          else
            if f=8 or f=9 or f=15 or f=16 then
              if c=11 or c=21 then
                c_color <= "100";
              else
                c_color <= "111";
              end if;
```

```
            else
              if f=10 or f=12 or f=14 then
                if c=11 or c=21 then
                  c_color <= "100";
                else
                  if c=15 or c=17 then
                    c_color <= "000";
                  else
                    c_color <= "111";
                  end if;
                end if;
              else
                if f=11 or f=13 then
                  if c=11 or c=21 then
                    c_color <= "100";
                  else
                    if c>=14 and c<=18 then
                      c_color <= "000";
                    else
                      c_color <= "111";
                    end if;
                  end if;
                else
                  if f>17 then
                    c_color <= "111";
                  end if;
                end if;
              end if;
            end if;
          end if;
        end if;
      when "01111011" => --tecla KP - ([7B]).
        if f<=6 then
          c_color <= "111";
        else
          if f=7 or f=17 then
            if c>=11 and c<=21 then
              c_color <= "100";
            else
              c_color <= "111";
            end if;
          else
            if f=8 or f=9 or f=10 or f=11 or f=13 or f=14 or f=15 or f=16 then
              if c=11 or c=21 then
                c_color <= "100";
              else
                c_color <= "111";
              end if;
            else
              if f=12 then
                if c=11 or c=21 then
                  c_color <= "100";
                else
                  if c>=15 and c<=17 then
                    c_color <= "000";
                  else
                    c_color <= "111";
                  end if;
                end if;
              else
                if f>17 then
                  c_color <= "111";
                end if;
              end if;
            end if;
          end if;
        end if;
      when "01111001" => --tecla KP + ([79]).
        if f<=6 then
          c_color <= "111";
        else
          if f=7 or f=17 then
            if c>=11 and c<=21 then
```

```
            c_color <= "100";
          else
            c_color <= "111";
          end if;
        else
          if f=8 or f=9 or f=15 or f=16 then
            if c=11 or c=21 then
              c_color <= "100";
            else
              c_color <= "111";
            end if;
          else
            if f=10 or f=11 or f=13 or f=14 then
              if c=11 or c=21 then
                c_color <= "100";
              else
                if c=16 then
                  c_color <= "000";
                else
                  c_color <= "111";
                end if;
              end if;
            else
              if f=12 then
                if c=11 or c=21 then
                  c_color <= "100";
                else
                  if c>=14 and c<=18 then
                    c_color <= "000";
                  else
                    c_color <= "111";
                  end if;
                end if;
              else
                if f>17 then
                  c_color <= "111";
                end if;
              end if;
            end if;
          end if;
        end if;
      end if;
    when "01110000" =>  --tecla KP 0([70]).
      if f<=6 then
        c_color <= "111";
      else
        if f=7 or f= 17 then
          if c>=11 and c<=21 then
            c_color <= "100";
          else
            c_color <= "111";
          end if;
        else
          if f=8 or f=16 then
            if c=11 or c=21 then
              c_color <= "100";
            else
              c_color <= "111";
            end if;
          else
            if f=9 or f=15 then
              if c=11 or c=21 then
                c_color <= "100";
              else
                if c>=14 and c<=18 then
                  c_color <= "000";
                else
                  c_color <= "111";
                end if;
              end if;
            else
              if f>=10 and f<=14 then
                if c=11 or c=21 then
```

```
                c_color <= "100";
              else
                if c=14 or c=18 then
                  c_color <= "000";
                else
                  c_color <= "111";
                end if;
              end if;
            else
              if f>17 then
                c_color <= "111";
              end if;
            end if;
          end if;
        end if;
      end if;
    end if;
  when "01101001" => --tecla KP 1([69]).
    if f<=6 then
      c_color <= "111";
    else
      if f=7 or f=17 then
        if c>=11 and c<=21 then
          c_color <= "100";
        else
          c_color <= "111";
        end if;
      else
        if f=8 or f=16 then
          if c=11 or c=21 then
            c_color <= "100";
          else
            c_color <= "111";
          end if;
        else
          if f=9 or f=11 or f=12 or f=13 or f=14 then
            if c=21 or c=11 then
              c_color <= "100";
            else
              if c=16 then
                c_color <= "000";
              else
                c_color <= "111";
              end if;
            end if;
          else
            if f=10 then
              if c=21 or c=11 then
                c_color <= "100";
              else
                if c=15 or c=16 then
                  c_color <= "000";
                else
                  c_color <= "111";
                end if;
              end if;
            else
              if f=15 then
                if c=21 or c=11 then
                  c_color <= "100";
                else
                  if c>=15 and c<=17 then
                    c_color <= "000";
                  else
                    c_color <= "111";
                  end if;
                end if;
              else
                if f>17 then
                  c_color <= "111";
                end if;
              end if;
            end if;
```

```
            end if;
          end if;
        end if;
      end if;
    when "01111010" => --tecla KP 3([7A]).
      if f<=6 then
        c_color <= "111";
      else
        if f=7 or f=17 then
          if c>=11 and c<=21 then
            c_color <= "100";
          else
            c_color <= "111";
          end if;
        else
          if f=8 or f=16 then
            if c=11 or c=21 then
              c_color <= "100";
            else
              c_color <= "111";
            end if;
          else
            if f=9 or f=12 or f=15 then
              if c=21 or c=11 then
                c_color <= "100";
              else
                if c>=14 and c<=18 then
                  c_color <= "000";
                else
                  c_color <= "111";
                end if;
              end if;
            else
              if f=10 or f=11 or f=13 or f=14 then
                if c=21 or c=11 then
                  c_color <= "100";
                else
                  if c=18 then
                    c_color <= "000";
                  else
                    c_color <= "111";
                  end if;
                end if;
              else
                if f>17 then
                  c_color <= "111";
                end if;
              end if;
            end if;
          end if;
        end if;
      end if;
    when "01110011" => --tecla KP 5([73]).
      if f<=6 then
        c_color <= "111";
      else
        if f=7 or f=17 then
          if c>=11 and c<=21 then
            c_color <= "100";
          else
            c_color <= "111";
          end if;
        else
          if f=8 or f=16 then
            if c=11 or c=21 then
              c_color <= "100";
            else
              c_color <= "111";
            end if;
          else
            if f=9 or f=12 or f=15 then
              if c=21 or c=11 then
                c_color <= "100";
```

```
              else
                if c>=14 and c<=18 then
                  c_color <= "000";
                else
                  c_color <= "111";
                end if;
              end if;
            else
              if f=10 or f=11 then
                if c=21 or c=11 then
                  c_color <= "100";
                else
                  if c=14 then
                    c_color <= "000";
                  else
                    c_color <= "111";
                  end if;
                end if;
              else
                if f=13 or f=14 then
                  if c=21 or c=11 then
                    c_color <= "100";
                  else
                    if c=18 then
                      c_color <= "000";
                    else
                      c_color <= "111";
                    end if;
                  end if;
                else
                  if f>17 then
                    c_color <= "111";
                  end if;
                end if;
              end if;
            end if;
          end if;
        end if;
      when "01101100" => --tecla KP 7([6C]).
        if f<=6 then
          c_color <= "111";
        else
          if f=7 or f=17 then
            if c>=11 and c<=21 then
              c_color <= "100";
            else
              c_color <= "111";
            end if;
          else
            if f=8 or f=16 then
              if c=11 or c=21 then
                c_color <= "100";
              else
                c_color <= "111";
              end if;
            else
              if f=9 or f=12 then
                if c=21 or c=11 then
                  c_color <= "100";
                else
                  if c>=14 and c<=18 then
                    c_color <= "000";
                  else
                    c_color <= "111";
                  end if;
                end if;
              else
                if f=10 or f=11 then
                  if c=21 or c=11 then
                    c_color <= "100";
                  else
                    if c=18 then
```

```
                    c_color <= "000";
                  else
                    c_color <= "111";
                  end if;
                end if;
              else
                if f=13 or f=14 or f=15 then
                  if c=21 or c=11 then
                    c_color <= "100";
                  else
                    if c=18 then
                      c_color <= "000";
                    else
                      c_color <= "111";
                    end if;
                  end if;
                else
                  if f>17 then
                    c_color <= "111";
                  end if;
                end if;
              end if;
            end if;
          end if;
        end if;
      when "01111101" => --tecla KP 9([7D]).
        if f<=6 then
          c_color <= "111";
        else
          if f=7 or f=17 then
            if c>=11 and c<=21 then
              c_color <= "100";
            else
              c_color <= "111";
            end if;
          else
            if f=8 or f=16 then
              if c=11 or c=21 then
                c_color <= "100";
              else
                c_color <= "111";
              end if;
            else
              if f=9 or f=12 or f=15 then
                if c=21 or c=11 then
                  c_color <= "100";
                else
                  if c>=14 and c<=18 then
                    c_color <= "000";
                  else
                    c_color <= "111";
                  end if;
                end if;
              else
                if f=10 or f=11 then
                  if c=21 or c=11 then
                    c_color <= "100";
                  else
                    if c=14 or c=18 then
                      c_color <= "000";
                    else
                      c_color <= "111";
                    end if;
                  end if;
                else
                  if f=13 or f=14 then
                    if c=21 or c=11 then
                      c_color <= "100";
                    else
                      if c=18 then
                        c_color <= "000";
                      else
```

```
                    c_color <= "111";
                  end if;
                end if;
              else
                if f>17 then
                  c_color <= "111";
                end if;
              end if;
            end if;
          end if;
        end if;
      end if;
    end if;
  when "00011100" => --tecla A([1C]).
    if f<=6 then
      c_color <= "111";
    else
      if f=7 or f=17 then
        if c>= 11 and c<=21 then
          c_color <= "100";
        else
          c_color <= "111";
        end if;
      else
        if f=8 or f=16 then
          if c=11 or c= 21 then
            c_color <= "100";
          else
            c_color <= "111";
          end if;
        else
          if f=9 then
            if c=21 or c=11 then
              c_color <= "100";
            else
              if c>=15 and c<=17 then
                c_color <= "000";
              else
                c_color <= "111";
              end if;
            end if;
          else
            if f=10 or f=11 or f=13 or f=14 or f= 15 then
              if c=21 or c=11 then
                c_color <= "100";
              else
                if c=14 or c=18 then
                  c_color <= "000";
                else
                  c_color <= "111";
                end if;
              end if;
            else
              if f=12 then
                if c=21 or c=11 then
                  c_color <= "100";
                else
                  if c>=14 and c<=18 then
                    c_color <= "000";
                  else
                    c_color <= "111";
                  end if;
                end if;
              else
                if f>17 then
                  c_color <= "111";
                end if;
              end if;
            end if;
          end if;
        end if;
      end if;
    end if;
```

```
      when "00110010" => --tecla B([32]).
        if f<=6 then
          c_color <= "111";
        else
          if f=7 or f=17 then
            if c>=11 and c<=21 then
              c_color <= "100";
            else
              c_color <= "111";
            end if;
          else
            if f=8 or f=16 then
              if c=11 or c=21 then
                c_color <= "100";
              else
                c_color <= "111";
              end if;
            else
              if f=9 or f=12 or f=15 then
                if c=21 or c=11 then
                  c_color <= "100";
                else
                  if c>=14 and c<=17 then
                    c_color <= "000";
                  else
                    c_color <= "111";
                  end if;
                end if;
              else
                if f=10 or f=11 or f=13 or f=14 then
                  if c=21 or c=11 then
                    c_color <= "100";
                  else
                    if c=14 or c=18 then
                      c_color <= "000";
                    else
                      c_color <= "111";
                    end if;
                  end if;
                else
                  if f>17 then
                    c_color <= "111";
                  end if;
                end if;
              end if;
            end if;
          end if;
        end if;
      when "00100001" => --tecla C([21]).
        if f<=6 then
          c_color <= "111";
        else
          if f=7 or f=17 then
            if c>=11 and c<=21 then
              c_color <= "100";
            else
              c_color <= "111";
            end if;
          else
            if f=8 or f=16 then
              if c=11 or c=21 then
                c_color <= "100";
              else
                c_color <= "111";
              end if;
            else
              if f=9 or f=15 then
                if c=11 or c=21 then
                  c_color <= "100";
                else
                  if c>=15 and c<=17 then
                    c_color <= "000";
                  else
```

```
                c_color <= "111";
              end if;
            end if;
          else
            if f=10 or f=14 then
              if c=11 or c=21 then
                c_color <= "100";
              else
                if c=14 or c=18 then
                  c_color <= "000";
                else
                  c_color <= "111";
                end if;
              end if;
            else
              if f>=11 and f<=13 then
                if c=11 or c=21 then
                  c_color <= "100";
                else
                  if c=14 then
                    c_color <= "000";
                  else
                    c_color <= "111";
                  end if;
                end if;
              else
                if f>17 then
                  c_color <= "111";
                end if;
              end if;
            end if;
          end if;
        end if;
      end if;
    when "00100011" =>  --tecla D([23]).
      if f<=6 then
        c_color <= "111";
      else
        if f=7 or f=17 then
          if c>=11 and c<=21 then
            c_color <= "100";
          else
            c_color <= "111";
          end if;
        else
          if f=16 or f=8 then
            if c=11 or c=21 then
              c_color <= "100";
            else
              c_color <= "111";
            end if;
          else
            if f=15 or f=9 then
              if c<=10 or c>=22 then
                c_color <= "111";
              else
                if c=11 or c=21 then
                  c_color <= "100";
                else
                  if c=14 or c=15 or c=16 or c=17 then
                    c_color <= "000";
                  else
                    c_color <= "111";
                  end if;
                end if;
              end if;
            else
              if f<=14 and f>=10 then
                if c<=10 or c>=22 then
                  c_color <= "111";
                else
                  if c=11 or c=21 then
```

```
                    c_color <= "100";
                  else
                    if c=14 or c=18 then
                      c_color <= "000";
                    else
                      c_color <= "111";
                    end if;
                  end if;
                end if;
              else
                if f>17 then
                  c_color <= "111";
                end if;
              end if;
            end if;
          end if;
        end if;
      end if;
    when "00100100" => --tecla E([24]).
      if f<=6 then
        c_color <= "111";
      else
        if f=7 or f=17 then
          if c>=11 and c<=21 then
            c_color <= "100";
          else
            c_color <= "111";
          end if;
        else
          if f=8 or f=16 then
            if c=11 or c=21 then
              c_color <= "100";
            else
              c_color <= "111";
            end if;
          else
            if f=9 or f=12 or f=15 then
              if c=21 or c=11 then
                c_color <= "100";
              else
                if c>=14 and c<=18 then
                  c_color <= "000";
                else
                  c_color <= "111";
                end if;
              end if;
            else
              if f=10 or f=11 or f=13 or f=14 then
                if c=21 or c=11 then
                  c_color <= "100";
                else
                  if c=14 then
                    c_color <= "000";
                  else
                    c_color <= "111";
                  end if;
                end if;
              else
                if f>17 then
                  c_color <= "111";
                end if;
              end if;
            end if;
          end if;
        end if;
      end if;
    when "00101011" => --tecla F([2B]).
      if f<=6 then
        c_color <= "111";
      else
        if f=7 or f=17 then
          if c>=11 and c<=21 then
            c_color <= "100";
```

```
            else
              c_color <= "111";
            end if;
          else
            if f=8 or f=16 then
              if c=11 or c=21 then
                c_color <= "100";
              else
                c_color <= "111";
              end if;
            else
              if f=9 or f=12 then
                if c=21 or c=11 then
                  c_color <= "100";
                else
                  if c>=14 and c<=18 then
                    c_color <= "000";
                  else
                    c_color <= "111";
                  end if;
                end if;
              else
                if f=10 or f=11 or f=13 or f=14 or f=15 then
                  if c=21 or c=11 then
                    c_color <= "100";
                  else
                    if c=14 then
                      c_color <= "000";
                    else
                      c_color <= "111";
                    end if;
                  end if;
                else
                  if f>17 then
                    c_color <= "111";
                  end if;
                end if;
              end if;
            end if;
          end if;
        end if;
      when "00110100" => --tecla G([34]).
        if f<=6 then
          c_color <= "111";
        else
          if f=7 or f=17 then
            if c>=11 and c<=21 then
              c_color <= "100";
            else
              c_color <= "111";
            end if;
          else
            if f=8 or f=16 then
              if c=11 or c=21 then
                c_color <= "100";
              else
                c_color <= "111";
              end if;
            else
              if f=9 or f=15 then
                if c=11 or c=21 then
                  c_color <= "100";
                else
                  if c>=15 and c<=17 then
                    c_color <= "000";
                  else
                    c_color <= "111";
                  end if;
                end if;
              else
                if f=10 or f=14 then
                  if c=11 or c=21 then
                    c_color <= "100";
```

```
                else
                  if c=14 or c=18 then
                    c_color <= "000";
                  else
                    c_color <= "111";
                  end if;
                end if;
              else
                if f>=11 and f<=12 then
                  if c=11 or c=21 then
                    c_color <= "100";
                  else
                    if c=14 then
                      c_color <= "000";
                    else
                      c_color <= "111";
                    end if;
                  end if;
                else
                  if f=13 then
                    if c=11 or c=21 then
                      c_color <= "100";
                    else
                      if c=14 or c=17 or c=18 then
                        c_color <= "000";
                      else
                        c_color <= "111";
                      end if;
                    end if;
                  else
                    if f>17 then
                      c_color <= "111";
                    end if;
                  end if;
                end if;
              end if;
            end if;
          end if;
        end if;
      end if;
    when "00110011" => --tecla H([33]).
      if f<=6 then
        c_color <= "111";
      else
        if f=7 or f=17 then
          if c>=11 and c<=21 then
            c_color <= "100";
          else
            c_color <= "111";
          end if;
        else
          if f=8 or f=16 then
            if c=11 or c=21 then
              c_color <= "100";
            else
              c_color <= "111";
            end if;
          else
            if f=9 or f=10 or f=11 or f=13 or f=14 or f=15 then
              if c=21 or c=11 then
                c_color <= "100";
              else
                if c=14 or c=18 then
                  c_color <= "000";
                else
                  c_color <= "111";
                end if;
              end if;
            else
              if f=12 then
                if c=21 or c=11 then
                  c_color <= "100";
                else
```

```
                if c>=14 and c<=18 then
                  c_color <= "000";
                else
                  c_color <= "111";
                end if;
              end if;
            else
              if f>17 then
                c_color <= "111";
              end if;
            end if;
          end if;
        end if;
      end if;
    when "01000011" => --tecla I([43]).
      if f<=6 then
        c_color <= "111";
      else
        if f=7 or f=17 then
          if c>=11 and c<=21 then
            c_color <= "100";
          else
            c_color <= "111";
          end if;
        else
          if f=8 or f=16 then
            if c=11 or c=21 then
              c_color <= "100";
            else
              c_color <= "111";
            end if;
          else
            if f=9 or f=15 then
              if c=21 or c=11 then
                c_color <= "100";
              else
                if c>=15 and c<=17 then
                  c_color <= "000";
                else
                  c_color <= "111";
                end if;
              end if;
            else
              if f>=10 and f<=14 then
                if c=21 or c=11 then
                  c_color <= "100";
                else
                  if c=16 then
                    c_color <= "000";
                  else
                    c_color <= "111";
                  end if;
                end if;
              else
                if f>17 then
                  c_color <= "111";
                end if;
              end if;
            end if;
          end if;
        end if;
      end if;
    when "00111011" => --tecla J([3B]).
      if f<=6 then
        c_color <= "111";
      else
        if f=7 or f=17 then
          if c>=11 and c<=21 then
            c_color <= "100";
          else
            c_color <= "111";
          end if;
```

```
          else
            if f=8 or f=16 then
              if c=11 or c=21 then
                c_color <= "100";
              else
                c_color <= "111";
              end if;
            else
              if f=9 then
                if c=21 or c=11 then
                  c_color <= "100";
                else
                  if c>=16 and c<=18 then
                    c_color <= "000";
                  else
                    c_color <= "111";
                  end if;
                end if;
              else
                if f>=10 and f<=12 then
                  if c=21 or c=11 then
                    c_color <= "100";
                  else
                    if c=17 then
                      c_color <= "000";
                    else
                      c_color <= "111";
                    end if;
                  end if;
                else
                  if f=13 or f=14 then
                    if c=21 or c=11 then
                      c_color <= "100";
                    else
                      if c=14 or c=17 then
                        c_color <= "000";
                      else
                        c_color <= "111";
                      end if;
                    end if;
                  else
                    if f=15 then
                      if c=21 or c=11 then
                        c_color <= "100";
                      else
                        if c>=15 and c<=16 then
                          c_color <= "000";
                        else
                          c_color <= "111";
                        end if;
                      end if;
                    else
                      if f>17 then
                        c_color <= "111";
                      end if;
                    end if;
                  end if;
                end if;
              end if;
            end if;
          end if;
        end if;
      when "01000010" => --tecla K([42]).
        if f<=6 then
          c_color <= "111";
        else
          if f=7 or f=17 then
            if c>=11 and c<=21 then
              c_color <= "100";
            else
              c_color <= "111";
            end if;
          else
```

```
            if f=8 or f=16 then
              if c=11 or c= 21 then
                c_color <= "100";
              else
                c_color <= "111";
              end if;
            else
              if f=9 or f=15 then
                if c=21 or c=11 then
                  c_color <= "100";
                else
                  if c=14 or c=18 then
                    c_color <= "000";
                  else
                    c_color <= "111";
                  end if;
                end if;
              else
                if f=10 or f=14 then
                  if c=21 or c=11 then
                    c_color <= "100";
                  else
                    if c=14 or c=17 then
                      c_color <= "000";
                    else
                      c_color <= "111";
                    end if;
                  end if;
                else
                  if f=12 then
                    if c=21 or c=11 then
                      c_color <= "100";
                    else
                      if c>=14 and c<=15 then
                        c_color <= "000";
                      else
                        c_color <= "111";
                      end if;
                    end if;
                  else
                    if f=13 or f=11 then
                      if c=21 or c=11 then
                        c_color <= "100";
                      else
                        if c=14 or c=16 then
                          c_color <= "000";
                        else
                          c_color <= "111";
                        end if;
                      end if;
                    else
                      if f>17 then
                        c_color <= "111";
                      end if;
                    end if;
                  end if;
                end if;
              end if;
            end if;
          end if;
        end if;
      when "01001011" => --tecla L([4B]).
        if f<=6 then
          c_color <= "111";
        else
          if f=7 or f=17 then
            if c>=11 and c<=21 then
              c_color <= "100";
            else
              c_color <= "111";
            end if;
          else
            if f=8 or f=16 then
```

```
            if c=11 or c=21 then
              c_color <= "100";
            else
              c_color <= "111";
            end if;
          else
            if f>=9 and f<=14 then
              if c=21 or c=11 then
                c_color <= "100";
              else
                if c=14 then
                  c_color <= "000";
                else
                  c_color <= "111";
                end if;
              end if;
            else
              if f=15 then
                if c=21 or c=11 then
                  c_color <= "100";
                else
                  if c>=14 and c<=18 then
                    c_color <= "000";
                  else
                    c_color <= "111";
                  end if;
                end if;
              else
                if f>17 then
                  c_color <= "111";
                end if;
              end if;
            end if;
          end if;
        end if;
      end if;
    when "00111010" => --tecla M([3A]).
      if f<=6 then
        c_color <= "111";
      else
        if f=7 or f=17 then
          if c>=11 and c<=21 then
            c_color <= "100";
          else
            c_color <= "111";
          end if;
        else
          if f=8 or f=16 then
            if c=11 or c=21 then
              c_color <= "100";
            else
              c_color <= "111";
            end if;
          else
            if f=9 or f=13 or f=14 or f=15 then
              if c=21 or c=11 then
                c_color <= "100";
              else
                if c=14 or c=18 then
                  c_color <= "000";
                else
                  c_color <= "111";
                end if;
              end if;
            else
              if f=10 then
                if c=21 or c=11 then
                  c_color <= "100";
                else
                  if c=14 or c=15 or c=17 or c=18 then
                    c_color <= "000";
                  else
                    c_color <= "111";
```

```
                end if;
              end if;
            else
              if f=11 then
                if c=21 or c=11 then
                  c_color <= "100";
                else
                  if c>=14 and c<=18 then
                    c_color <= "000";
                  else
                    c_color <= "111";
                  end if;
                end if;
              else
                if f=12 then
                  if c=21 or c=11 then
                    c_color <= "100";
                  else
                    if c=14 or c=16 or c=18 then
                      c_color <= "000";
                    else
                      c_color <= "111";
                    end if;
                  end if;
                else
                  if f>17 then
                    c_color <= "111";
                  end if;
                end if;
              end if;
            end if;
          end if;
        end if;
      end if;
    end if;
  when "00110001" => --tecla N([31]).
    if f<=6 then
      c_color <= "111";
    else
      if f=7 or f=17 then
        if c>=11 and c<=21 then
          c_color <= "100";
        else
          c_color <= "111";
        end if;
      else
        if f=8 or f=16 then
          if c=11 or c=21 then
            c_color <= "100";
          else
            c_color <= "111";
          end if;
        else
          if f=9 or f=15 then
            if c=21 or c=11 then
              c_color <= "100";
            else
              if c=14 or c=18 then
                c_color <= "000";
              else
                c_color <= "111";
              end if;
            end if;
          else
            if f=10 then
              if c=21 or c=11 then
                c_color <= "100";
              else
                if c=14 or c=15 or c=18 then
                  c_color <= "000";
                else
                  c_color <= "111";
                end if;
```

```
                end if;
              else
                if f=11 then
                  if c=21 or c=11 then
                    c_color <= "100";
                  else
                    if c=14 or c=15 or c=16 or c=18 then
                      c_color <= "000";
                    else
                      c_color <= "111";
                    end if;
                  end if;
                else
                  if f=12 then
                    if c=21 or c=11 then
                      c_color <= "100";
                    else
                      if c=14 or c=16 or c=18 then
                        c_color <= "000";
                      else
                        c_color <= "111";
                      end if;
                    end if;
                  else
                    if f=13 then
                      if c=21 or c=11 then
                        c_color <= "100";
                      else
                        if c=14 or c=16 or c=17 or c=18 then
                          c_color <= "000";
                        else
                          c_color <= "111";
                        end if;
                      end if;
                    else
                      if f=14 then
                        if c=21 or c=11 then
                          c_color <= "100";
                        else
                          if c=14 or c=17 or c=18 then
                            c_color <= "000";
                          else
                            c_color <= "111";
                          end if;
                        end if;
                      else
                        if f>17 then
                          c_color <= "111";
                        end if;
                      end if;
                    end if;
                  end if;
                end if;
              end if;
            end if;
          end if;
        end if;
      end if;
    when "01001100" =>  --tecla Ñ([4C]).
      if f<=6 then
        c_color <= "111";
      else
        if f=7 or f=17 then
          if c>=11 and c<=21 then
            c_color <= "100";
          else
            c_color <= "111";
          end if;
        else
          if f=8 or f=16 or f=10 then
            if c=11 or c=21 then
              c_color <= "100";
            else
```

```
              c_color <= "111";
            end if;
          else
            if f=9 then
              if c=21 or c=11 then
                c_color <= "100";
              else
                if c>=14 and c<=18 then
                  c_color <= "000";
                else
                  c_color <= "111";
                end if;
              end if;
            else
              if f=11 or f=15 then
                if c=21 or c=11 then
                  c_color <= "100";
                else
                  if c=14 or c=18 then
                    c_color <= "000";
                  else
                    c_color <= "111";
                  end if;
                end if;
              else
                if f=12 then
                  if c=21 or c=11 then
                    c_color <= "100";
                  else
                    if c=14 or c=15 or c=18 then
                      c_color <= "000";
                    else
                      c_color <= "111";
                    end if;
                  end if;
                else
                  if f=13 then
                    if c=21 or c=11 then
                      c_color <= "100";
                    else
                      if c=14 or c=16 or c=18 then
                        c_color <= "000";
                      else
                        c_color <= "111";
                      end if;
                    end if;
                  else
                    if f=14 then
                      if c=21 or c=11 then
                        c_color <= "100";
                      else
                        if c=14 or c=17 or c=18 then
                          c_color <= "000";
                        else
                          c_color <= "111";
                        end if;
                      end if;
                    else
                      if f>17 then
                        c_color <= "111";
                      end if;
                    end if;
                  end if;
                end if;
              end if;
            end if;
          end if;
        end if;
      end if;
    when "01000100" =>  --tecla O([44]).
      if f<=6 then
        c_color <= "111";
      else
```

```
          if f=7 or f= 17 then
            if c>=11 and c<=21 then
              c_color <= "100";
            else
              c_color <= "111";
            end if;
          else
            if f=8 or f=16 then
              if c=11 or c=21 then
                c_color <= "100";
              else
                c_color <= "111";
              end if;
            else
              if f=9 or f=15 then
                if c=11 or c=21 then
                  c_color <= "100";
                else
                  if c>=15 and c<=17 then
                    c_color <= "000";
                  else
                    c_color <= "111";
                  end if;
                end if;
              else
                if f>=10 and f<=14 then
                  if c=11 or c=21 then
                    c_color <= "100";
                  else
                    if c=14 or c=18 then
                      c_color <= "000";
                    else
                      c_color <= "111";
                    end if;
                  end if;
                else
                  if f>17 then
                    c_color <= "111";
                  end if;
                end if;
              end if;
            end if;
          end if;
        end if;
      when "01001101" => --tecla P([4D]).
        if f<=6 then
          c_color <= "111";
        else
          if f=7 or f= 17 then
            if c>=11 and c<=21 then
              c_color <= "100";
            else
              c_color <= "111";
            end if;
          else
            if f=8 or f=16 then
              if c=11 or c=21 then
                c_color <= "100";
              else
                c_color <= "111";
              end if;
            else
              if f=9 or f=12 then
                if c=21 or c=11 then
                  c_color <= "100";
                else
                  if c>=14 and c<=17 then
                    c_color <= "000";
                  else
                    c_color <= "111";
                  end if;
                end if;
              else
```

```
              if f=10 or f=11 then
                if c=21 or c=11 then
                  c_color <= "100";
                else
                  if c=14 or c=18 then
                    c_color <= "000";
                  else
                    c_color <= "111";
                  end if;
                end if;
              else
                if f=13 or f=14 or f=15 then
                  if c=21 or c=11 then
                    c_color <= "100";
                  else
                    if c=14 then
                      c_color <= "000";
                    else
                      c_color <= "111";
                    end if;
                  end if;
                else
                  if f>17 then
                    c_color <= "111";
                  end if;
                end if;
              end if;
            end if;
          end if;
        end if;
      end if;
    when "00010101" =>  --tecla Q([15]).
      if f<=6 then
        c_color <= "111";
      else
        if f=7 or f= 17 then
          if c>=11 and c<=21 then
            c_color <= "100";
          else
            c_color <= "111";
          end if;
        else
          if f=8 or f=16 then
            if c=11 or c=21 then
              c_color <= "100";
            else
              c_color <= "111";
            end if;
          else
            if f=9 then
              if c=11 or c=21 then
                c_color <= "100";
              else
                if c>=15 and c<=17 then
                  c_color <= "000";
                else
                  c_color <= "111";
                end if;
              end if;
            else
              if f=10 or f=11 or f=12 or f=14 then
                if c=11 or c=21 then
                  c_color <= "100";
                else
                  if c=14 or c=18 then
                    c_color <= "000";
                  else
                    c_color <= "111";
                  end if;
                end if;
              else
                if f=13 then
                  if c=11 or c=21 then
```

```
                    c_color <= "100";
                  else
                    if c=14 or c=17 or c=18 then
                      c_color <= "000";
                    else
                      c_color <= "111";
                    end if;
                  end if;
                else
                  if f=15 then
                    if c=11 or c=21 then
                      c_color <= "100";
                    else
                      if c=15 or c=16 or c=17 or c=19 then
                        c_color <= "000";
                      else
                        c_color <= "111";
                      end if;
                    end if;
                  else
                    if f>17 then
                      c_color <= "111";
                    end if;
                  end if;
                end if;
              end if;
            end if;
          end if;
        end if;
      end if;
    end if;
  when "00101101" => --tecla R([2D]).
    if f<=6 then
      c_color <= "111";
    else
      if f=7 or f= 17 then
        if c>=11 and c<=21 then
          c_color <= "100";
        else
          c_color <= "111";
        end if;
      else
        if f=8 or f=16 then
          if c=11 or c=21 then
            c_color <= "100";
          else
            c_color <= "111";
          end if;
        else
          if f=9 or f=12 then
            if c=21 or c=11 then
              c_color <= "100";
            else
              if c>=14 and c<=17 then
                c_color <= "000";
              else
                c_color <= "111";
              end if;
            end if;
          else
            if f=10 or f=11 or f=13 or f=14 or f=15 then
              if c=21 or c=11 then
                c_color <= "100";
              else
                if c=14 or c=18 then
                  c_color <= "000";
                else
                  c_color <= "111";
                end if;
              end if;
            else
              if f>17 then
                c_color <= "111";
              end if;
```

```
            end if;
          end if;
        end if;
      end if;
    end if;
  when "00011011" => --tecla S([1B]).
    if f<=6 then
      c_color <= "111";
    else
      if f=7 or f= 17 then
        if c>=11 and c<=21 then
          c_color <= "100";
        else
          c_color <= "111";
        end if;
      else
        if f=8 or f=16 then
          if c=11 or c=21 then
            c_color <= "100";
          else
            c_color <= "111";
          end if;
        else
          if f=9 or f=12 or f=15 then
            if c=21 or c=11 then
              c_color <= "100";
            else
              if c>=15 and c<=17 then
                c_color <= "000";
              else
                c_color <= "111";
              end if;
            end if;
          else
            if f=10 or f=14 then
              if c=21 or c=11 then
                c_color <= "100";
              else
                if c=14 or c=18 then
                  c_color <= "000";
                else
                  c_color <= "111";
                end if;
              end if;
            else
              if f=11 then
                if c=21 or c=11 then
                  c_color <= "100";
                else
                  if c=14 then
                    c_color <= "000";
                  else
                    c_color <= "111";
                  end if;
                end if;
              else
                if f=13 then
                  if c=21 or c=11 then
                    c_color <= "100";
                  else
                    if c=18 then
                      c_color <= "000";
                    else
                      c_color <= "111";
                    end if;
                  end if;
                else
                  if f>17 then
                    c_color <= "111";
                  end if;
                end if;
              end if;
            end if;
```

```
            end if;
          end if;
        end if;
      end if;
    when "00101100" => --tecla T([2C]).
      if f<=6 then
        c_color <= "111";
      else
        if f=7 or f= 17 then
          if c>=11 and c<=21 then
            c_color <= "100";
          else
            c_color <= "111";
          end if;
        else
          if f=8 or f=16 then
            if c=11 or c=21 then
              c_color <= "100";
            else
              c_color <= "111";
            end if;
          else
            if f=9 then
              if c=21 or c=11 then
                c_color <= "100";
              else
                if c>=14 and c<=18 then
                  c_color <= "000";
                else
                  c_color <= "111";
                end if;
              end if;
            else
              if f>=10 and f<=15 then
                if c=21 or c=11 then
                  c_color <= "100";
                else
                  if c=16 then
                    c_color <= "000";
                  else
                    c_color <= "111";
                  end if;
                end if;
              else
                if f>17 then
                  c_color <= "111";
                end if;
              end if;
            end if;
          end if;
        end if;
      end if;
    when "00111100" => --tecla U([3C]).
      if f<=6 then
        c_color <= "111";
      else
        if f=7 or f= 17 then
          if c>=11 and c<=21 then
            c_color <= "100";
          else
            c_color <= "111";
          end if;
        else
          if f=8 or f=16 then
            if c=11 or c=21 then
              c_color <= "100";
            else
              c_color <= "111";
            end if;
          else
            if f>=9 and f<=14 then
              if c=21 or c=11 then
                c_color <= "100";
```

```
              else
                if c=14 or c=18 then
                  c_color <= "000";
                else
                  c_color <= "111";
                end if;
              end if;
            else
              if f=15 then
                if c=21 or c=11 then
                  c_color <= "100";
                else
                  if c>=15 and c<=17 then
                    c_color <= "000";
                  else
                    c_color <= "111";
                  end if;
                end if;
              else
                if f>17 then
                  c_color <= "111";
                end if;
              end if;
            end if;
          end if;
        end if;
      end if;
    when "00101010" => --tecla V([2A]).
      if f<=6 then
        c_color <= "111";
      else
        if f=7 or f= 17 then
          if c>=11 and c<=21 then
            c_color <= "100";
          else
            c_color <= "111";
          end if;
        else
          if f=8 or f=16 then
            if c=11 or c=21 then
              c_color <= "100";
            else
              c_color <= "111";
            end if;
          else
            if f>=9 and f<=12 then
              if c=21 or c=11 then
                c_color <= "100";
              else
                if c=14 or c=18 then
                  c_color <= "000";
                else
                  c_color <= "111";
                end if;
              end if;
            else
              if f=13 or f=14 then
                if c=21 or c=11 then
                  c_color <= "100";
                else
                  if c=15 or c=17 then
                    c_color <= "000";
                  else
                    c_color <= "111";
                  end if;
                end if;
              else
                if f=15 then
                  if c=21 or c=11 then
                    c_color <= "100";
                  else
                    if c=16 then
                      c_color <= "000";
```

```
                    else
                      c_color <= "111";
                    end if;
                  end if;
                else
                  if f>17 then
                    c_color <= "111";
                  end if;
                end if;
              end if;
            end if;
          end if;
        end if;
      when "00011101" => --tecla W([1D]).
        if f<=6 then
          c_color <= "111";
        else
          if f=7 or f= 17 then
            if c>=11 and c<=21 then
              c_color <= "100";
            else
              c_color <= "111";
            end if;
          else
            if f=8 or f=16 then
              if c=11 or c=21 then
                c_color <= "100";
              else
                c_color <= "111";
              end if;
            else
              if f=9 or f=10 or f=11 or f=15 then
                if c=21 or c=11 then
                  c_color <= "100";
                else
                  if c=14 or c=18 then
                    c_color <= "000";
                  else
                    c_color <= "111";
                  end if;
                end if;
              else
                if f=12 then
                  if c=21 or c=11 then
                    c_color <= "100";
                  else
                    if c=14 or c=16 or c=18 then
                      c_color <= "000";
                    else
                      c_color <= "111";
                    end if;
                  end if;
                else
                  if f=13 then
                    if c=21 or c=11 then
                      c_color <= "100";
                    else
                      if c>=14 and c<=18 then
                        c_color <= "000";
                      else
                        c_color <= "111";
                      end if;
                    end if;
                  else
                    if f=14 then
                      if c=21 or c=11 then
                        c_color <= "100";
                      else
                        if c=14 or c=15 or c=17 or c=18 then
                          c_color <= "000";
                        else
                          c_color <= "111";
```

```
                    end if;
                  end if;
                else
                  if f>17 then
                    c_color <= "111";
                  end if;
                end if;
              end if;
            end if;
          end if;
        end if;
      end if;
    when "00100010" => --tecla X([22]).
      if f<=6 then
        c_color <= "111";
      else
        if f=7 or f= 17 then
          if c>=11 and c<=21 then
            c_color <= "100";
          else
            c_color <= "111";
          end if;
        else
          if f=8 or f=16 then
            if c=11 or c=21 then
              c_color <= "100";
            else
              c_color <= "111";
            end if;
          else
            if f=9 or f=10 or f=14 or f=15 then
              if c=21 or c=11 then
                c_color <= "100";
              else
                if c=14 or c=18 then
                  c_color <= "000";
                else
                  c_color <= "111";
                end if;
              end if;
            else
              if f=11 or f=13 then
                if c=21 or c=11 then
                  c_color <= "100";
                else
                  if c=15 or c=17 then
                    c_color <= "000";
                  else
                    c_color <= "111";
                  end if;
                end if;
              else
                if f=12 then
                  if c=21 or c=11 then
                    c_color <= "100";
                  else
                    if c=16 then
                      c_color <= "000";
                    else
                      c_color <= "111";
                    end if;
                  end if;
                else
                  if f>17 then
                    c_color <= "111";
                  end if;
                end if;
              end if;
            end if;
          end if;
        end if;
      end if;
```

```
      when "00110101" => --tecla Y([35]).
        if f<=6 then
          c_color <= "111";
        else
          if f=7 or f= 17 then
            if c>=11 and c<=21 then
              c_color <= "100";
            else
              c_color <= "111";
            end if;
          else
            if f=8 or f=16 then
              if c=11 or c=21 then
                c_color <= "100";
              else
                c_color <= "111";
              end if;
            else
              if f=9 or f=10 then
                if c=21 or c=11 then
                  c_color <= "100";
                else
                  if c=14 or c=18 then
                    c_color <= "000";
                  else
                    c_color <= "111";
                  end if;
                end if;
              else
                if f=11 then
                  if c=21 or c=11 then
                    c_color <= "100";
                  else
                    if c=15 or c=17 then
                      c_color <= "000";
                    else
                      c_color <= "111";
                    end if;
                  end if;
                else
                  if f>=12 and f<=15 then
                    if c=21 or c=11 then
                      c_color <= "100";
                    else
                      if c=16 then
                        c_color <= "000";
                      else
                        c_color <= "111";
                      end if;
                    end if;
                  else
                    if f>17 then
                      c_color <= "111";
                    end if;
                  end if;
                end if;
              end if;
            end if;
          end if;
        end if;
      when "00011010" => --tecla Z([1A]).
        if f<=6 then
          c_color <= "111";
        else
          if f=7 or f= 17 then
            if c>=11 and c<=21 then
              c_color <= "100";
            else
              c_color <= "111";
            end if;
          else
            if f=8 or f=16 then
              if c=11 or c=21 then
```

```
            c_color <= "100";
          else
            c_color <= "111";
          end if;
        else
          if f=9 or f=15 then
            if c=21 or c=11 then
              c_color <= "100";
            else
              if c>=14 and c<=18 then
                c_color <= "000";
              else
                c_color <= "111";
              end if;
            end if;
          else
            if f=10 then
              if c=21 or c=11 then
                c_color <= "100";
              else
                if c=18 then
                  c_color <= "000";
                else
                  c_color <= "111";
                end if;
              end if;
            else
              if f=11 then
                if c=21 or c=11 then
                  c_color <= "100";
                else
                  if c=17 then
                    c_color <= "000";
                  else
                    c_color <= "111";
                  end if;
                end if;
              else
                if f=12 then
                  if c=21 or c=11 then
                    c_color <= "100";
                  else
                    if c=16 then
                      c_color <= "000";
                    else
                      c_color <= "111";
                    end if;
                  end if;
                else
                  if f=13 then
                    if c=21 or c=11 then
                      c_color <= "100";
                    else
                      if c=15 then
                        c_color <= "000";
                      else
                        c_color <= "111";
                      end if;
                    end if;
                  else
                    if f=14 then
                      if c=21 or c=11 then
                        c_color <= "100";
                      else
                        if c=14 then
                          c_color <= "000";
                        else
                          c_color <= "111";
                        end if;
                      end if;
                    else
                      if f>17 then
                        c_color <= "111";
```

```
                    end if;
                  end if;
                end if;
              end if;
            end if;
          end if;
        end if;
      end if;
    end if;
  when "01000101" =>  --tecla 0([45]).
    if f<=6 then
      c_color <= "111";
    else
      if f=7 or f= 17 then
        if c>=11 and c<=21 then
          c_color <= "100";
        else
          c_color <= "111";
        end if;
      else
        if f=8 or f=16 then
          if c=11 or c=21 then
            c_color <= "100";
          else
            c_color <= "111";
          end if;
        else
          if f=9 or f=15 then
            if c=11 or c=21 then
              c_color <= "100";
            else
              if c>=14 and c<=18 then
                c_color <= "000";
              else
                c_color <= "111";
              end if;
            end if;
          else
            if f>=10 and f<=14 then
              if c=11 or c=21 then
                c_color <= "100";
              else
                if c=14 or c=18 then
                  c_color <= "000";
                else
                  c_color <= "111";
                end if;
              end if;
            else
              if f>17 then
                c_color <= "111";
              end if;
            end if;
          end if;
        end if;
      end if;
    end if;
  when "00010110" => --tecla 1([16]).
    if f<=6 then
      c_color <= "111";
    else
      if f=7 or f=17 then
        if c>=11 and c<=21 then
          c_color <= "100";
        else
          c_color <= "111";
        end if;
      else
        if f=8 or f=16 then
          if c=11 or c=21 then
            c_color <= "100";
          else
```

```
              c_color <= "111";
            end if;
          else
            if f=9 or f=11 or f=12 or f=13 or f=14 then
              if c=21 or c=11 then
                c_color <= "100";
              else
                if c=16 then
                  c_color <= "000";
                else
                  c_color <= "111";
                end if;
              end if;
            else
              if f=10 then
                if c=21 or c=11 then
                  c_color <= "100";
                else
                  if c=15 or c=16 then
                    c_color <= "000";
                  else
                    c_color <= "111";
                  end if;
                end if;
              else
                if f=15 then
                  if c=21 or c=11 then
                    c_color <= "100";
                  else
                    if c>=15 and c<=17 then
                      c_color <= "000";
                    else
                      c_color <= "111";
                    end if;
                  end if;
                else
                  if f>17 then
                    c_color <= "111";
                  end if;
                end if;
              end if;
            end if;
          end if;
        end if;
      end if;
    when "00011110" => --tecla 2([1E]).
      if f<=6 then
        c_color <= "111";
      else
        if f=7 or f=17 then
          if c>=11 and c<=21 then
            c_color <= "100";
          else
            c_color <= "111";
          end if;
        else
          if f=8 or f=16 then
            if c=11 or c=21 then
              c_color <= "100";
            else
              c_color <= "111";
            end if;
          else
            if f=9 or f=12 or f=15 then
              if c=21 or c=11 then
                c_color <= "100";
              else
                if c>=14 and c<=18 then
                  c_color <= "000";
                else
                  c_color <= "111";
                end if;
              end if;
```

```
            else
              if f=10 or f=11 then
                if c=21 or c=11 then
                  c_color <= "100";
                else
                  if c=18 then
                    c_color <= "000";
                  else
                    c_color <= "111";
                  end if;
                end if;
              else
                if f=13 or f=14 then
                  if c=21 or c=11 then
                    c_color <= "100";
                  else
                    if c=14 then
                      c_color <= "000";
                    else
                      c_color <= "111";
                    end if;
                  end if;
                else
                  if f>17 then
                    c_color <= "111";
                  end if;
                end if;
              end if;
            end if;
          end if;
        end if;
      end if;
    when "00100110" => --tecla 3([26]).
      if f<=6 then
        c_color <= "111";
      else
        if f=7 or f=17 then
          if c>=11 and c<=21 then
            c_color <= "100";
          else
            c_color <= "111";
          end if;
        else
          if f=8 or f=16 then
            if c=11 or c=21 then
              c_color <= "100";
            else
              c_color <= "111";
            end if;
          else
            if f=9 or f=12 or f=15 then
              if c=21 or c=11 then
                c_color <= "100";
              else
                if c>=14 and c<=18 then
                  c_color <= "000";
                else
                  c_color <= "111";
                end if;
              end if;
            else
              if f=10 or f=11 or f=13 or f=14 then
                if c=21 or c=11 then
                  c_color <= "100";
                else
                  if c=18 then
                    c_color <= "000";
                  else
                    c_color <= "111";
                  end if;
                end if;
              else
                if f>17 then
```

```
                c_color <= "111";
              end if;
            end if;
          end if;
        end if;
      end if;
    end if;
  when "00100101" => --tecla 4([25]).
    if f<=6 then
      c_color <= "111";
    else
      if f=7 or f=17 then
        if c>=11 and c<=21 then
          c_color <= "100";
        else
          c_color <= "111";
        end if;
      else
        if f=8 or f=16 then
          if c=11 or c=21 then
            c_color <= "100";
          else
            c_color <= "111";
          end if;
        else
          if f=9 or f=10 or f=11 or f=12 then
            if c=21 or c=11 then
              c_color <= "100";
            else
              if c=14 or c=18 then
                c_color <= "000";
              else
                c_color <= "111";
              end if;
            end if;
          else
            if f=13 then
              if c=21 or c=11 then
                c_color <= "100";
              else
                if c>=14 and c<=18 then
                  c_color <= "000";
                else
                  c_color <= "111";
                end if;
              end if;
            else
              if f=14 or f=15 then
                if c=21 or c=11 then
                  c_color <= "100";
                else
                  if c=18 then
                    c_color <= "000";
                  else
                    c_color <= "111";
                  end if;
                end if;
              else
                if f>17 then
                  c_color <= "111";
                end if;
              end if;
            end if;
          end if;
        end if;
      end if;
    end if;
  when "00101110" => --tecla 5([2E]).
    if f<=6 then
      c_color <= "111";
    else
      if f=7 or f=17 then
        if c>=11 and c<=21 then
```

```
            c_color <= "100";
          else
            c_color <= "111";
          end if;
        else
          if f=8 or f=16 then
            if c=11 or c=21 then
              c_color <= "100";
            else
              c_color <= "111";
            end if;
          else
            if f=9 or f=12 or f=15 then
              if c=21 or c=11 then
                c_color <= "100";
              else
                if c>=14 and c<=18 then
                  c_color <= "000";
                else
                  c_color <= "111";
                end if;
              end if;
            else
              if f=10 or f=11 then
                if c=21 or c=11 then
                  c_color <= "100";
                else
                  if c=14 then
                    c_color <= "000";
                  else
                    c_color <= "111";
                  end if;
                end if;
              else
                if f=13 or f=14 then
                  if c=21 or c=11 then
                    c_color <= "100";
                  else
                    if c=18 then
                      c_color <= "000";
                    else
                      c_color <= "111";
                    end if;
                  end if;
                else
                  if f>17 then
                    c_color <= "111";
                  end if;
                end if;
              end if;
            end if;
          end if;
        end if;
      end if;
    when "00110110" => --tecla 6([36]).
      if f<=6 then
        c_color <= "111";
      else
        if f=7 or f=17 then
          if c>=11 and c<=21 then
            c_color <= "100";
          else
            c_color <= "111";
          end if;
        else
          if f=8 or f=16 then
            if c=11 or c=21 then
              c_color <= "100";
            else
              c_color <= "111";
            end if;
          else
            if f=9 or f=12 or f=15 then
```

```
              if c=21 or c=11 then
                c_color <= "100";
              else
                if c>=14 and c<=18 then
                  c_color <= "000";
                else
                  c_color <= "111";
                end if;
              end if;
            else
              if f=10 or f=11 then
                if c=21 or c=11 then
                  c_color <= "100";
                else
                  if c=14 then
                    c_color <= "000";
                  else
                    c_color <= "111";
                  end if;
                end if;
              else
                if f=13 or f=14 then
                  if c=21 or c=11 then
                    c_color <= "100";
                  else
                    if c=14 or c=18 then
                      c_color <= "000";
                    else
                      c_color <= "111";
                    end if;
                  end if;
                else
                  if f>17 then
                    c_color <= "111";
                  end if;
                end if;
              end if;
            end if;
          end if;
        end if;
      end if;
    when "00111101" => --tecla 7([3D]).
      if f<=6 then
        c_color <= "111";
      else
        if f=7 or f=17 then
          if c>=11 and c<=21 then
            c_color <= "100";
          else
            c_color <= "111";
          end if;
        else
          if f=8 or f=16 then
            if c=11 or c=21 then
              c_color <= "100";
            else
              c_color <= "111";
            end if;
          else
            if f=9 or f=12 then
              if c=21 or c=11 then
                c_color <= "100";
              else
                if c>=14 and c<=18 then
                  c_color <= "000";
                else
                  c_color <= "111";
                end if;
              end if;
            else
              if f=10 or f=11 then
                if c=21 or c=11 then
                  c_color <= "100";
```

```
                else
                  if c=18 then
                    c_color <= "000";
                  else
                    c_color <= "111";
                  end if;
                end if;
              else
                if f=13 or f=14 or f=15 then
                  if c=21 or c=11 then
                    c_color <= "100";
                  else
                    if c=18 then
                      c_color <= "000";
                    else
                      c_color <= "111";
                    end if;
                  end if;
                else
                  if f>17 then
                    c_color <= "111";
                  end if;
                end if;
              end if;
            end if;
          end if;
        end if;
      end if;
    when "00111110" => --tecla 8([3E]).
      if f<=6 then
        c_color <= "111";
      else
        if f=7 or f=17 then
          if c>=11 and c<=21 then
            c_color <= "100";
          else
            c_color <= "111";
          end if;
        else
          if f=8 or f=16 then
            if c=11 or c=21 then
              c_color <= "100";
            else
              c_color <= "111";
            end if;
          else
            if f=9 or f=12 or f=15 then
              if c=21 or c=11 then
                c_color <= "100";
              else
                if c>=14 and c<=18 then
                  c_color <= "000";
                else
                  c_color <= "111";
                end if;
              end if;
            else
              if f=10 or f=11 or f=13 or f=14 then
                if c=21 or c=11 then
                  c_color <= "100";
                else
                  if c=14 or c=18 then
                    c_color <= "000";
                  else
                    c_color <= "111";
                  end if;
                end if;
              else
                if f>17 then
                  c_color <= "111";
                end if;
              end if;
            end if;
```

```
          end if;
        end if;
      end if;
    when "01000110" => --tecla 9([46]).
      if f<=6 then
        c_color <= "111";
      else
        if f=7 or f=17 then
          if c>=11 and c<=21 then
            c_color <= "100";
          else
            c_color <= "111";
          end if;
        else
          if f=8 or f=16 then
            if c=11 or c=21 then
              c_color <= "100";
            else
              c_color <= "111";
            end if;
          else
            if f=9 or f=12 or f=15 then
              if c=21 or c=11 then
                c_color <= "100";
              else
                if c>=14 and c<=18 then
                  c_color <= "000";
                else
                  c_color <= "111";
                end if;
              end if;
            else
              if f=10 or f=11 then
                if c=21 or c=11 then
                  c_color <= "100";
                else
                  if c=14 or c=18 then
                    c_color <= "000";
                  else
                    c_color <= "111";
                  end if;
                end if;
              else
                if f=13 or f=14 then
                  if c=21 or c=11 then
                    c_color <= "100";
                  else
                    if c=18 then
                      c_color <= "000";
                    else
                      c_color <= "111";
                    end if;
                  end if;
                else
                  if f>17 then
                    c_color <= "111";
                  end if;
                end if;
              end if;
            end if;
          end if;
        end if;
      end if;
    WHEN OTHERS => --(marco rojo en el centro de la pantalla para imprimir letras).
      if f<=6 then
        c_color <= "111";
      else
        if f=7 or f=17 then
          if c>=11 and c<=21 then
            c_color <= "100";
          else
            c_color <= "111";
          end if;
```

```
          else
            if f<=16 and f>=8 then
              if c=11 or c=21 then
                c_color <= "100";
              else
                c_color <= "111";
              end if;
            else
              if f>17 then
                c_color <= "111";
              end if;
            end if;
          end if;
        end if;
    END CASE;
  end if;
end process PRUEBA;
end BEH;
```

16.4. Código *VHDL* del Top Level.

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_ARITH.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;
entity TOP_LEVEL is
  port(clk      : in std_logic;--reloj principal de la fpga (50MHz)->T=20ns.
       rst      : in std_logic;--reinicio global del sistema.
       ps2_clk  : in std_logic;--reloj del puerto PS/2.
       ps2_dato : in std_logic;--datos del puerto PS/2.
       HSync    : out std_logic;--señal de sincronismo horizontal activa a nivel bajo de
                                --1 bit, que se activa a los 656 ciclos de reloj.
       VSync    : out std_logic;--señal de sincronismo vertical activa a nivel bajo de 1
                                --bit, que se activa a los 416 771 ciclos de reloj.
       R,G,B    : out std_logic);--señales de 1 bit que sirven para seleccionar el color
                                 --a sacar por pantalla.
end TOP_LEVEL;
architecture BEH of TOP_LEVEL is
-------------Declaracion de señales-----------------
signal a_clk,a_rst  : std_logic;
signal error,c_error: std_logic;
signal X,Y,c_X,c_Y  : std_logic_vector(9 downto 0);--señal de 10 bits (2^10=1024)
signal tecla,c_tecla: std_logic_vector(7 downto 0);--codigo hexadecimal de la tecla
                                                   --presionada.
signal Color,c_color: std_logic_vector(2 downto 0);--señal de 3 bits que indica el
                                                   --código del color a dar al píxel que
                                                   --se esté procesando.
-------------Declaracion de componentes-------------
component VGA is
  port(clk   : in std_logic;--reloj principal de la fpga (50MHz)->T=20ns.
       rst   : in std_logic;--reinicio global del sistema.
       Color : in std_logic_vector(2 downto 0);--señal de 3 bits que indica el código
                                               --del color a dar al píxel que se esté
                                               --procesando.
       HSync : out std_logic;--señal de sincronismo horizontal activa a nivel bajo de 1
                             --bit, que se activa a los 656 ciclos de reloj.
       VSync : out std_logic;--señal de sincronismo vertical activa a nivel bajo de 1
                             --bit, que se activa a los 416 771 ciclos de reloj.
       R,G,B : out std_logic;--señales de 1 bit que sirven para seleccionar el color a
                             --sacar por pantalla.
       X     : out std_logic_vector(9 downto 0);--señal de 10 bits (2^10=1024) para
                                                --alcanzar los 640 pixeles de ancho.
       Y     : out std_logic_vector(9 downto 0));--señal de 10 bits (2^10=1024) para
                                                 --alcanzar los 480 pixeles de alto.
end component;
component teclado107 is
  port(ps2_clk   : in  std_logic;--reloj del puerto PS/2.
       ps2_dato  : in  std_logic;--datos del puerto PS/2.
       clk       : in  std_logic;--reloj principal de la fpga (50MHz)->T=20ns.
       rst       : in  std_logic;--reinicio global del sistema.
       error     : out std_logic;--señal de salida de error.
       tecla     : out std_logic_vector(7 downto 0));--codigo hexadecimal de la tecla
                                                     --presionada.
end component;
component Controlador is
  port(clk     : in std_logic;--reloj principal de la fpga (50MHz)->T=20ns.
       rst     : in std_logic;--reinicio global del sistema.
       c_tecla : in std_logic_vector(7 downto 0);--codigo hexadecimal de la tecla
                                                 --presionada.
       c_error : in std_logic;--señal de salida de error.
       c_X     : in std_logic_vector(9 downto 0);--señal de 10 bits (2^10=1024) para
                                                 --alcanzar los 640 pixeles de ancho.
       c_Y     : in std_logic_vector(9 downto 0);--señal de 10 bits (2^10=1024) para
                                                 --alcanzar los 480 pixeles de alto.
       c_color : out std_logic_vector(2 downto 0));--señal de 3 bits que indica el
                                                   --código del Color a dar al píxel que
                                                   --se esté procesando.
end component;
----Asignacion y conexionado entre bloques: teclado PS/2, pantalla VGA y controlador----
begin
  c_tecla <= tecla;
  c_error <= error;
```

```
  c_X     <= X;
  c_Y     <= Y;
  Color   <= c_color;
  a_clk   <= clk;
  a_rst   <= rst;
-----------Mapeo de componentes------
teclado: teclado107
port map(ps2_clk  => ps2_clk,
         ps2_dato => ps2_dato,
         clk      => a_clk,
         rst      => a_rst,
         error    => error,
         tecla    => tecla);
pantalla: VGA
port map(clk   => a_clk,
         rst   => a_rst,
         Color => Color,
         HSync => HSync,
         VSync => VSync,
         R     => R,
         G     => G,
         B     => B,
         X     => X,
         Y     => Y);
conductor: Controlador
port map(clk     => a_clk,
         rst     => a_rst,
         c_tecla => c_tecla,
         c_error => c_error,
         c_X     => c_X,
         c_Y     => c_Y,
         c_color => c_color);

end BEH;
```

17. Bibliografía y Referencias.

1. **Xilinx.** Xcell Issue 32, Second Quarter. *Xilinx Inc.* [En línea] 1999. [Citado el: 20 de diciembre de 2013.] http://www.xilinx.com/publications/archives/xcell/Xcell32.pdf.
2. **Kaczynski, Jerry. Technical Manager, Aldec Inc.** The Challenges of Modern FPGA Design Verification. *EE Journal.* [En línea] 27 de 07 de 2004. [Citado el: 20 de diciembre de 2013.] http://www.eejournal.com/archives/articles/20040727_aldec/.
3. **National Instruments.** FPGAs a Fondo. *National Instruments Corporation.* [En línea] 20 de junio de 2012. [Citado el: 20 de diciembre de 2013.] http://www.ni.com/white-paper/6983/es/.
4. **Xilinx.** All Programmable FPGAs. *Xilinx Inc.* [En línea] 2013. [Citado el: 20 de diciembre de 2013.] http://www.xilinx.com/products/silicon-devices/fpga/index.htm.
5. **Xilinx .** MicroBlaze Soft Processor Core. *Xilinx Inc.* [En línea] 2013. [Citado el: 20 de diciembre de 2013.] http://www.xilinx.com/tools/microblaze.htm.
6. **Xilinx.** PicoBlaze 8-bit Microcontroller. *Xilinx Inc.* [En línea] 2013. [Citado el: 20 de diciemrbre de 2013.] http://www.xilinx.com/products/intellectual-property/picoblaze.htm.
7. **Altera.** Nios II Processor: The World's Most Versatile Embedded Processor. *Altera Corporation.* [En línea] 2013. [Citado el: 20 de diciembre de 2013.] http://www.altera.com/devices/processor/nios2/ni2-index.html.
8. **ARM.** Processors - ARM. *ARM Ltd.* [En línea] 2013. [Citado el: 20 de diciembre de 2013.] http://www.arm.com/products/processors/index.php.
9. **Xilinx.** Partial Reconfiguration in the ISE Design Suite. *Xilinx Inc.* [En línea] 2013. [Citado el: 20 de diciembre de 2013.] http://www.xilinx.com/tools/partial-reconfiguration.htm.
10. **Altera.** About Partial Reconfiguration. *Altera Corporation.* [En línea] 2005. [Citado el: 20 de diciembre de 2013.] http://quartushelp.altera.com/13.1/master.htm#mergedProjects/comp/comp/comp_about_part_reconfig.htm?GSA_pos=1&WT.oss_r=1&WT.oss=partial%20reconfiguration.
11. **Xilinx.** ISE Design Suite. *Xilinx Inc.* [En línea] 2013. [Citado el: 20 de diciembre de 2013.] http://www.xilinx.com/products/design-tools/ise-design-suite/index.htm.
12. **IBM.** The first 10 years. *IBM Corporation.* [En línea] 2013. [Citado el: 20 de diciembre de 2013.] http://www-03.ibm.com/ibm/history/exhibits/pc25/pc25_tenyears.html.
13. **Biography, Dictionary of Wisconsin.** Sholes, Christopher Latham 1819 - 1890. *Wisconsin Historical Society.* [En línea] 1996-2013. [Citado el: 20 de diciembre de 2013.] http://www.wisconsinhistory.org/dictionary/index.asp?action=view&term_id=1741&keyword=sholes.
14. **Allcalmap.** Christopher Latham Sholes. *Find A Grave.* [En línea] 05 de julio de 2003. [Citado el: 20 de diciembre de 2013.] http://www.findagrave.com/cgi-bin/fg.cgi?page=gr&GRid=7656870.
15. **IBM.** The birth of the IBM PC. *IBM Corporation.* [En línea] 2013. [Citado el: 20 de diciembre de 2013.] http://www-03.ibm.com/ibm/history/exhibits/pc25/pc25_birth.html.
16. **IBM.** Introduction to keyboards. *IBM Corporation.* [En línea] 2013. [Citado el: 20 de diciembre de 2013.] http://www-01.ibm.com/software/globalization/topics/keyboards/physical.html.
17. **Cassingham, Randy.** The Dvorak Keyboard. *ThisisTrue Inc.* [En línea] abril de 2012. [Citado el: 20 de diciembre de 2013.] http://www.dvorak-keyboard.com/.
18. **MediaWiki.** FAQ. *Colemak Keyboard Layout.* [En línea] 17 de marzo de 2010. [Citado el: 20 de diciembre de 2013.] http://colemak.com/FAQ.
19. **IBM.** Chronological History of IBM 1980s. *IBM Corporation.* [En línea] 2013. [Citado el: 20 de diciembre de 2013.] http://www-03.ibm.com/ibm/history/history/decade_1980.html.
20. **ISO/IEC.** Consolidated text of ISO/IEC 9995, Keyboard layouts for text and office systems. *INTERNATIONAL ORGANIZATION FOR STANDARDIZATION/INTERNATIONAL ELECTROTECHNICAL COMMISSION.* [En línea] 15 de febrero de 2001. [Citado el: 20 de diciembre de 2013.] http://web.archive.org/web/20070719075251/http:/cyberiel.iquebec.com/sc35wg1/sc35n0232_9995-2.pdf.
21. **IBM.** ISO9995 Standard. *IBM Corporation.* [En línea] 2013. [Citado el: 20 de diciembre de 2013.] http://www-01.ibm.com/software/globalization/topics/keyboards/iso.html.
22. **S. Thompson.** VGA-sign choices for a new video subsystem. *IBM Corporation.* [En línea] 1988. [Citado el: 20 de diciembre de 2013.]

http://domino.research.ibm.com/tchjr/journalindex.nsf/d9f0a910ab8b637485256bc80066a393/c7d637c81ed36da585256bfa00685beb!OpenDocument.
23. **Chapweske, Adam.** The PS/2 Mouse/Keyboard Protocol. *Computer-Engineering.org.* [En línea] 05 de septiembre de 2003. [Citado el: 20 de diciembre de 2013.] http://www.computer-engineering.org/ps2protocol/.
24. **Xilinx Inc.** Chapter 8-PS/2 Mouse/Keyboard Port, page. 63. *Spartan-3E FPGA Starter Kit Board User Guide-UG230 (v1.2).* [En línea] 20 de enero de 2011. [Citado el: 20 de diciembre de 2013.] http://www.xilinx.com/support/documentation/boards_and_kits/ug230.pdf.
25. **Chapweske, Adam.** The PS/2 Keyboard Interface. *Computer-Engineering.org.* [En línea] 04 de enero de 2003. [Citado el: 20 de diciembre de 2013.] http://www.computer-engineering.org/ps2keyboard/.
26. **Xilinx Inc.** Chapter 6-VGA Display Port, page. 55. *Spartan-3E FPGA Starter Kit Board User Guide-UG230 (v1.2).* [En línea] 20 de enero de 2011. [Citado el: 20 de diciembre de 2013.] http://www.xilinx.com/support/documentation/boards_and_kits/ug230.pdf.
27. **Departamento de Tecnología Electrónica, Universidad Carlos III de Madrid, Escuela Politécnica Superior.** *Manual de Prácticas de la asignatura Laboratorio de Microelectrónica de Ingeniería de Telecomunicación.* Madrid, Leganes : s.n., 25/09/2009. Práctica 1 - 4º Curso, 1er Cuatrimestre.
28. **Smith, D.J.** *HDL Chip Design.* s.l. : Doone, 1997. 9780965193436.